CLUBE

Z

Abril/84

N.º 19

Coordenadores: MARIA IRENE • ALEXANDRE SOUSA • J. MAGALHÃES • AV. BOAVISTA, 832-2.º T. • TELEF. 65127 • 4100 PORTO

NESTE NÚMERO

Por limitação de espaço, só no próximo número iniciaremos uma nova rúbrica de FERNANDO PRECES: **CONVERSÃO DE PROGRAMAS DO ZX81 → SPECTRUM**

No Interior:

Cupão de Inscrição

Edição: Clube Z80

Fotocomposição: Fotomecânica Mabreu/Porto

Impressão: Ramos dos Santos & C.ª, Lda./Porto

Tiragem: 500 exemplares, Abril 1984

INTRODUÇÃO À LINGUAGEM MÁQUINA

ZX81/SPECTRUM

Autor: FERNANDO PRECES

(Cont. dos números anteriores)

PARTE II — SISTEMAS DE NUMERAÇÃO

2.1. — Introdução

Hoje, como no passado, o homem na sua infância aprende a contar pelos dedos. É lógico que o seu cérebro utilize sempre, como base de cálculo, o sistema decimal e que oponha resistência ou sinta dificuldade no tratamento de sistemas numéricos noutra base de agrupamento.
No entanto, a mais poderosa máquina por ele criada até hoje (o computador digital), não pode, por razões tecnológicas, **pensar,** dentro dum sistema de base 10, sem o auxílio dum programa-monitor especializado, que adapte a linguagem do homem e o seu sistema numérico à **sua** linguagem.
Essa adaptação, porém, é paga por um alto preço: a lentidão de resposta; a ocupação indiscriminada de milhares de bytes de memória; e a limitação de algumas funções da máquina.
Para ultrapassar tal situação, deve o utilizador mais experimentado libertar-se, tanto quanto possível, de uma parte importante das instruções de tais linguagens, e tentar estabelecer contacto com a máquina numa linguagem semi-directa, indicada pelo fabricante, no respectivo manual. E é aqui que surge a necessidade de sabermos trabalhar com outros sistemas de numeração.

2.2. — Sistema Binário

Os computadores digitais usam um sistema de numeração na base 2, vulgarmente chamado "sistema binário". Utilizam, portanto, uma concepção numérica com apenas 2 algarismos: ("0" e "1"). Os sinais digitais por eles elaborados, são o resultado dos dois estados que os seus circuitos podem discriminar.
Suponha o leitor que vai ensaiar um painel com 4 lâmpadas ligadas em paralelo a uma fonte de tensão adequada, isoladas cada uma por um interruptor, tal como mostra a figura 2.1.

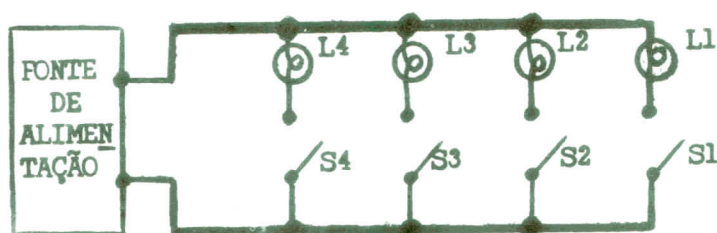



FIGURA 2.1. — Painel com 4 lâmpadas ligadas em paralelo a uma fonte de tensão

A quantidade de informação, resultante da manipulação dos 4 interruptores, é a seguinte:

	Em Binário	Em Decimal
4 lâmpadas apagadas	0000	0
L_1 acesa	0001	1
L_2 acesa	0010	2
L_1 e L_2 acesas	0011	3
L_3 acesa	0100	4
L_3 e L_1 acesas	0101	5
L_3 e L_2 acesas	0110	6
L_3, L_2 e L_1 acesas	0111	7
L_4 acesa	1000	8
L_4 e L_1 acesas	1001	9
L_4 e L_2 acesas	1010	10
L_4, L_2 e L_1 acesas	1011	11
L_4 e L_3 acesas	1100	12
L_4, L_3 e L_1 acesas	1101	13
L_4, L_3 e L_2 acesas	1110	14
4 lâmpadas acesas	1111	15

Conclui-se deste ensaio que as 4 lâmpadas podem indicar **16** situações diferentes, numeradas à direita em decimal (0 a 15).
Depois deste exemplo, poderá o leitor facilmente concluir que, se ligar nesse painel 8 lâmpadas pelo mesmo processo, vai conseguir encontrar **256** permutações possíveis, que poderá numerar em decimal de (0 a 255). Assim:

com 4 lâmpadas (16 combinações) 0000 a 1111
com 8 lâmpadas (256 combinações) 00000000 a 11111111

No sistema decimal, as posições que os algarismos ocupam na formação dum número, têm o peso de:

$10^0 = 1$ (para a unidade)
$10^1 = 10$ (» » dezena)
$10^2 = 100$ (» » centena)
$10^3 = 1000$ (» o milhar)
etc.

No sistema binário, os pesos são:

$2^0 = 1$ (para o bit 1)
$2^1 = 2$ (» » » 2)
$2^2 = 4$ (» » » 3)
$2^3 = 8$ (» » » 4)
$2^4 = 16$ (» » » 5)
etc.

Para converter o binário para decimal, tendo em conta os pesos das posições dos diversos dígitos, basta realizar a operação abaixo exemplificada.

Vamos escolher um número binário ao acaso: 10110

Peso do dígito	2^4	2^3	2^2	2^1	2^0
Número:	1	0	1	1	0

$$1 * 2^4 + 0 * 2^3 + 1 * 2^2 + 1 * 2^1 + 0 * 2^0$$

$$= 16 + 4 + 2 = 22 \text{ (decimal)}$$

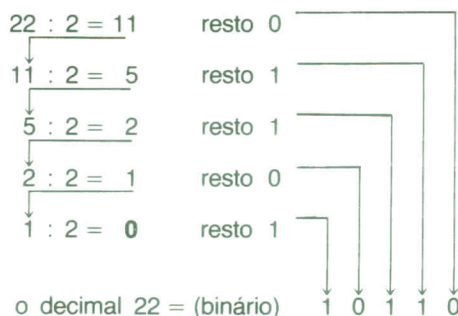
Eis um pequeno programa em Basic, tão aproximado quanto possível da operação acabada de realizar, que converte números binários em decimais.

Os possuidores do ZX81, terão apenas de substituir o símbolo ((↑) elevado a) por ((**) código 216), na linha 40.

```
 5 REM SISTEMAS DE NUMERAÇÃO — Programa 1
10 REM CONVERSÃO NÚMERO BINÁRIO EM DECIMAL
15 LET DECIMAL = 0
20 REM INTROD. NUM. BINÁRIO
25 INPUT B$
30 FOR P = LEN B$ TO 1 STEP - 1
40 LET D = (VAL B$ (P)) * (2 ↑ (LEN B$ - P))
50 LET DECIMAL = DECIMAL + D
60 NEXT P
80 PRINT AT 10,4; "DECIMAL = " ; DECIMAL
```

Para converter um número decimal num número binário, sem passar por outra base de numeração, o método é um pouco mais moroso. Vejamos o exemplo, aproveitando o número decimal da operação anterior.

(Decimal) 22 = (Binário) 10110

Vamos executar uma divisão sucessiva por 2. O quociente obtido na primeira divisão, volta a ser dividido por 2 e assim por diante, até ser alcançado um quociente 0.

Um pequeno programa em Basic que pode converter números decimais entre 0 e 65535 em números binários, auxiliará o leitor, efectuando as divisões acima referidas.

```
 100 REM CONVERSAO DE UM NUMERO
DECIMAL PARA BINARIO - PROGRAMA
2
 110 LET X=0
 115 LET B$="0000 0000 0000 0000
"
 120 REM INTROD. NUMERO DECIMAL
 125 INPUT N
 130 LET C=INT (N/2)
 140 LET D=N-2*C
 150 LET N=C
 160 LET B$(16-X)=STR$ D
 170 LET X=X+1
 175 IF N<>0 THEN GO TO 130
 180 PRINT AT 10,4;"BINARIO=";B$
```

NOTA: Trabalha nos 2 computadores ZX81 e Spectrum

Temos de reconhecer que, deixar de trabalhar num sistema com 10 algarismos, com o qual estamos totalmente identificados, começar com outro que tem apenas 2 algarismos, necessitando de **16 bits** para representar um número decimal de 5 dígitos, é realmente uma estopada de todo o tamanho. Mas a situação é ultrapassável se introduzirmos de permeio um sistema de numeração cuja base seja múltipla da base 2.

2.3. — Sistema numérico hexadecimal

Este sistema utiliza 16 símbolos diferentes; como o sistema decimal é composto por 10, os restantes são representados pelas seis primeiras letras do nosso alfabeto (ver fig. 2.2.).

Decimal	Hexadecimal	Binário
0	00	000000
1	01	000001
2	02	000010
3	03	000011
4	04	000100
5	05	000101
6	06	000110
7	07	000111
8	08	001000
9	09	001001
10	0A	001010
11	0B	001011
12	0C	001100
13	0D	001101
14	0E	001110
15	0F	001111
16	10	010000
17	11	010001
18	12	010010
19	13	010011
20	14	010100
21	15	010101
22	16	010110
23	17	010111
24	18	011000
25	19	011001
26	1A	011010
27	1B	011011
28	1C	011100
29	1D	011101
30	1E	011110
31	1F	011111
32	20	100000

FIGURA 2.2 — As 3 bases de numeração

A base 16 é múltipla à 4.ª potência, da base 2 (24 = 16), o que torna muito fácil a conversão de Binário a Hexadecimal. Para exemplo, tomemos o maior número binário de 8 bits, que iremos separar em grupos de 4.

1111	1111
2.° grupo	1.° grupo

Pelo quadro da figura 2.2, verifica-se que o Binário (1111), corresponde ao Hexa (F). Assim:

2.° grupo 1111 (b) = F (h)
1.° grupo 1111 (b) = F (h)

NOTA: O (b) quer dizer Binário e o (h) Hexadecimal.

Ao número 1111 1111 (b) corresponde o número FF (h).
Tal como fizemos com as outras bases de numeração, vamos gora considerar o **peso** de cada algoritmo do sistema Hexa.

$$16^0 = 1,\ 16^1 = 16,\ 16^2 = 256,\ 16^3 = 4096,\ \text{etc.}$$

O número FFFF (h), vai servir para exemplo na conversão a número decimal.

Peso ou posição	(h)	Operação	Resultado
(16^3) 4.º algoritmo	F	15 * 4096	61440
(16^2) 3.º »	F	15 * 256	3840
(16^1) 2.º »	F	15 * 16	240
(16^0) 1.º »	F	15 * 1	(+) 15
		Total →	65535

Para a passagem de decimal a hexa, vamos usar um processo idêntico ao utilizado pelos computadores para guardar os números.
O número 65535 vai servir como exemplo. A primeira operação consiste em dividi-lo por 256.

65535/256 = 255 • 9960975 . . .

Parte inteira (255) — Parte fraccionária (• 9960975 . . .)

Ao número encontrado como quociente, extrai-se de imediato a parte inteira (255), a que vamos chamar o byte mais significativo (HIGT BYTE). De seguida multiplica-se a parte fraccionária por 256.

• 9960975 . . . * 256 = 255

Ao número obtido na 2.ª operação, vamos chamar byte menos significativo (LOW BYTE).
Recorrendo ao quadro da fig. 2.3, que evita mais contas, encontramos facilmente a correspondência entre o decimal (255) e o hexa (FF). Substituindo os valores, temos:

FF (h) — HIGT BYTE; FF (h) — LOW BYTE

Resultando, depois da função:

FFFF (h) = 65535

Outro exemplo para praticarmos:

Decimal 32767

32767/256 = 127 • 9960975 . . .

• 9960975 . . . * 256 = 255

HIGT BYTE = 127, LOW BYTE = 255

127 (d) = 7F (h)
255 (d) = FF (h)

32767 (d) = 7F FF (h)

A passagem de hexa a binário, não oferece qualquer problema, visto que a cada algoritmo hexa, corresponde sempre um grupo de 4 bits em binário (ver quadro da figura 2.2).

1.º CHR \ 2.º CHR	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
2	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
4	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
5	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95
6	96	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111
7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Hex				
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
A	1	0	1	0
B	1	0	1	1
C	1	1	0	0
D	1	1	0	1
E	1	1	1	0
F	1	1	1	1

FIGURA 2.3 — *a)* **Conversão Hexa-decimal** — *b)* **Conversão Hexa-binário**

INTRODUÇÃO À LINGUAGEM MÁQUINA

RECAPITULAÇÃO DE JOGOS

FERNANDO PRECES

JOGOS DO FINAL DO 1.º CAPÍTULO

(v. n.º 15, Dezembro — págs. 2 e 3
n.º 16, Janeiro — págs. 2 a 5)

```
   1 REM E:RND?▄▀ TAN ▀?÷▀E(RND-
F/ RUN ( NEXT TAN XX
   3 GOSUB 1000
  10 LET S=0
  20 LET A=CODE "<"
  30 LET B=A
  50 LET Z=3
 100 FOR D=3 TO A*A
 110 PRINT AT B,Z;
 120 IF USR 16514=A THEN GOTO 30
0
 130 PRINT "O";
 140 IF INKEY$="F" THEN GOSUB 50
0
 190 LET B=B-(INKEY$="7" AND B)+
(INKEY$="6" AND B<A)
 200 PRINT AT RND*A,17;"<"
 210 LET C=USR 16521
 220 NEXT D
 300 PRINT "▒";S;" PONTOS"
 310 FOR T=0 TO 300
 320 NEXT T
 325 CLS
 330 PRINT AT 12,0;"QUER CONTINU
AR? (DIGA S OU N)."
 340 INPUT L$
 345 CLS
 350 IF L$="S" THEN GOTO 10
 355 PRINT AT 12,0;"ATE A PROXIM
A E... OBRIGADA.";W
 500 FOR C=Z TO PI
 510 IF USR 16514=A THEN LET S=S
+1
 520 PRINT "-";
 530 NEXT C
 540 PRINT AT B,Z;"O     "
 550 RETURN
1000 PRINT AT 1,5;"BATALHA NO ES
PACO"
1010 PRINT ,," A SUA NAVE E ATAC
ADA POR CENTE-"
1020 PRINT ,,"NAS DE PEQUENAS  N
AVES, CUJO FO-"
1030 PRINT ,,"GO E  INOFENSIVO M
AS,  PERIGOSAS"
1040 PRINT ,,"AO CHOQUE.  A TECL
A ""7"" DESVIA A"
1050 PRINT ,,"NAVE PARA CIMA, A
""6"" PARA BAIXO"
1060 PRINT ,,"E A TECLA ""F"" DI
SPARA."
1070 PRINT ,,"BOA SORTE..."
1080 PRINT ,,"PRIMA ""N/L"" PARA
 COMECAR."
1090 INPUT L$
1095 CLS
1099 RETURN
2000 SAVE "JOGO █"
2010 RUN
```

```
   5 REM JOGO 2
  10 REM PROGRAMA EM BASIC
  20 SLOW
  30 CLS
  40 GOSUB 100
  50 FOR A=1 TO 18
  60 PRINT AT A,0;"▒";AT A,31;"▒
  70 NEXT A
  80 GOSUB 100
  90 GOTO 140
 100 FOR A=0 TO 31
 110 PRINT "▒";
 120 NEXT A
 130 RETURN
 140 LET ED=1
 150 LET CB=1
 160 LET X=3
 170 LET Y=INT (RND*33+8)
 180 PLOT X,Y
 190 IF INKEY$="R" THEN RUN
 200 IF X=2 OR X=61 THEN LET ED=
-ED
 210 IF Y=6 OR Y=41 THEN LET CB=
-CB
 220 UNPLOT X,Y
 230 LET X=X+ED
 240 LET Y=Y+CB
 250 GOTO 180
 500 SAVE "JOGO █"
 510 RUN
```

```
  10 REM Y▒▄▀4NOT ( CLEAR TAN Y<█
?LN PRINT ▒Y▒NOT TAN LN F?LN E▀L
N ▙RND▄▀>VAL : LN ▒RNDAT VAL :3LN
 ▒RNDAT ( LIST LN ▙RNDTAN LN ▙RN
DTAN
  11 REM COPY ▀▄5Y▀ MWRNDM█RNDWWM
""RND GOSUB ?""RNDY█MKRNDLN █"LN
 █▀ FOR 5 CLS LOAD █ GOSUB ?COS
5█ CLEAR █ GOSUB ?COS U""RND RET
URN ▀C▄ RETURN X4▒U█RNDJWM█RNDUA
T RND RETURN ▄▀C▄ RETURN D4▒U█RND
JWM█RND GOSUB ?""RNDY█MKRNDLN █"
5█RND▀ ""RND▒█▀F▀▒█▀/█45678901345
67890
  20 LET K=USR 16533
  30 POKE 16577,INT (RND*33+8)
  35 PRINT AT 10,8;"▒▒▒▒▒▒▒▒"
  36 PLOT 22,22
  37 PLOT 25,19
  40 LET K=USR 16578
 100 REM JOGO 2 EM CODIGO MAQUIN
A
 500 SAVE "JOGO █"
 510 RUN
```

AOS LEITORES:

FERNANDO PRECES enviou-nos também uma listagem com o conjunto dos programas de todo o 1.º capítulo da rubrica INTRODUÇÃO À LINGUAGEM MÁQUINA, ZX81. Publicá-la-emos se os leitores nos manifestarem o seu interesse — ficamos a aguardar a v/ opinião.

TROCA DE PROGRAMAS

ANTÓNIO NUNES enviou ao CLUBE Z80 uma lista dos 76 programas que possui para trocar c/ sócios por outros que não tem. Não publicamos essa lista por ser muito extensa e os interessados podem pedi-la ao CLUBE, enviando 20$00 em selos CTT's, ou contactar c/ o sócio:

ANTÓNIO NUNES, Rua do Til, 72 — 9000 FUNCHAL.

ENCICLOPÉDIA BASIC — ABRIL/84

(Cont. dos números anteriores)

AUTO — comando

Este comando fornece um meio expedito de inserir números de linhas no programa já escrito. Quer a linha inicial, quer o intervalo entre números de linha, podem ser automáticos (por exemplo Hewllet Packard).

EXEMPLO:

AUTO 100,10 coloca o número inicial em 100 e o intervalo entre linhas, será de 10 em 10.

ESTE COMANDO É USADO NO INÍCIO OU NO DECORRER DA INTRODUÇÃO DO CÓDIGO BASIC, MAS NÃO DENTRO DE UM PROGRAMA.

BASE — instrução

Trata-se de uma instrução pertencente à norma ANSI e que é possível encontrar no Control Data ou no New Brain (option base).
A sua utilização destina-se a definir o valor mais baixo de um ARRAY (quadro ou tabela ou matriz), como 0 ou 1.

EXEMPLO:

```
10 BASE 0
20 DIM A(15)
```

A instrução BASE define o array desde A(0) até A(15), ou seja 16 elementos.
Muitos computadores estabelecem os arrays entre 0 e 10, ou seja 11 elementos, sem que exista essa declaração inicial.
A instrução BASE só pode ser usada antes do DIMensionamento e antes da manipulação das variáveis indexadas.
Alguns dialectos de BASIC permitem que a instrução BASE seja limite inferior das variáveis indexadas.

EXEMPLO:

```
10 BASE 6
20 DIM A(20)
```

definiria o array como 15 elementos, desde A(6) até A(20).

BREAK — comando

Trata-se de um comando que permite interromper o curso de um programa. Algumas máquinas permitem usar este comando no interior do programa, de modo a interromper o programa antes das linhas especificadas.

EXEMPLO:

10 BREAK 40,80,120 significa que o programa se detém antes das linhas 40, 80 e 120, de modo a que o operador possa verificar ou alterar determinadas condições. (De um modo similar ao STOP).
Regra geral, após uma interrupção originada por BREAK, poderá recomeçar, usando o comando CONtinue ou COntinue.
Os valores das variáveis não são afectados por este comando. Em muitas máquinas, o comando BREAK está implementado numa das teclas.

BYE — comando ou instrução

Este comando é usado frequentemente para sair do BASIC.
É um caso típico dos sistemas em TIME-SHARING, que aceitam o BYE como sinal do fim da execução.
NO CASO DAS MÁQUINAS ATARI — SOL por exemplo, o comando BYE serve para chamar o Sistema Operativo (disk operating system).
Em alternativa alguns sistemas permitem usar GOODBYE para significar fim da tarefa e SYSTEM para chamar o DOS.

(Cont. no próximo número)

ESTRATÉGIA

ZX81 16 K

Autor: FERNANDO PRECES
Sacavém

```
0 REM PROGRAMA MODIFICADO POR
ALMEIDA PRECES EM 20/7/1982
2 REM "5"
3 GOSUB 5000
4 INPUT T
5 CLS
6 IF T=1 THEN GOTO 8
7 IF T=2 THEN GOTO 2000
8 LET C=168
9 LET H=173
10 DIM A(25)
20 DIM B(7)
30 FOR A=3 TO 5
40 LET A(A)=C
50 NEXT A
70 FOR A=21 TO 23
80 LET A(A)=H
90 NEXT A
 110 LET B(1)=5
 120 LET B(2)=4
 130 LET B(3)=6
 140 LET B(4)=-4
 150 LET B(5)=-5
 160 LET B(6)=-6
 165 GOSUB 1000
 170 FOR A=20 TO 1 STEP -1
 175 FOR J=1 TO 5
 180 LET B=INT (RND*3)+1
 185 IF 5*INT (A/5)=A AND B=3 TH
EN GOTO 210
 190 IF A(A)=C AND A(A+B(B))=0 T
HEN GOTO 270
 200 NEXT J
 210 NEXT A
 220 FOR A=7 TO 20
 230 FOR B=4 TO 6
```

```
 235 IF 5*INT (A/5)=A AND B=4 TH
EN GOTO 250
 240 IF A(A)=C AND A(A+B(B))=0 T
HEN GOTO 270
 250 NEXT B
 260 NEXT A
 265 PRINT AT 20,0;"   EU PASSO
 "
 270 LET A(A+B(B))=C
 280 LET A(A)=0
 290 GOSUB 1000
 310 INPUT A$
 320 LET D=5*(CODE A$-38)+CODE A
$(2)-28
 330 LET E=5*(CODE A$(3)-38)+COD
E A$(4)-28
 340 LET A(E)=H
 350 LET A(D)=0
 360 GOTO 165
1000 LET X=0
1010 LET Y=0
1012 PRINT AT 20,0;"██████████
██████"
1013 LET Z=RND**RND
1015 PRINT AT 20,0;"------------
--------"
1020 PRINT AT 5,0;"1 2 3 4 5"
1025 PRINT
1030 FOR A=1 TO 25
1040 IF A(A)=0 THEN PRINT "▒ ";
1045 IF A(A)<>0 THEN PRINT CHR$
(A(A));" ";
1050 IF 5*INT (A/5)=A THEN PRINT
 " ";CHR$ (A/5+37),,,;
1060 IF A(A)=C AND A>20 THEN LET
 X=X+1
1070 IF A(A)=H AND A<6 THEN LET
Y=Y+1
1080 NEXT A
1090 PRINT "1 2 3 4 5"
1095 PRINT
1100 PRINT "HOMEM ";Y;" COMPUTAD
OR ";X
1101 IF Y>X THEN PRINT "VOCE EST
A GANHANDO"
1102 IF X>Y THEN PRINT "--EU EST
OU GANHANDO"
1105 IF X=4 OR Y=4 THEN GOTO 112
0
1110 RETURN
1120 IF X>Y THEN PRINT "COMPUTAD
OR";
1130 IF Y>X THEN PRINT "HOMEN";
1140 PRINT "VITORIA COM ";ABS (X
-Y);" PONTOS"
1150 STOP
2005 LET C=141
2007 LET H=140
2010 DIM A(25)
2020 DIM B(7)
2030 FOR A=3 TO 5
2040 LET A(A)=C
2050 NEXT A
2070 LET A(11)=H
2080 LET A(16)=H
2090 LET A(21)=H
2110 LET B(1)=5
2120 LET B(2)=4
2130 LET B(3)=6
2140 LET B(4)=-4
2150 LET B(5)=-5
2160 LET B(6)=-6
2165 GOSUB 3000
2170 FOR A=20 TO 1 STEP -1
2175 FOR J=1 TO 5
2180 LET B=INT (RND*3)+1
2185 IF 5*INT (A/5)=A AND B=3 TH
EN GOTO 2210
2190 IF A(A)=C AND A(A+B(B))=0 T
HEN GOTO 2270
2200 NEXT J
```

```
2210 NEXT A
2220 FOR A=7 TO 20
2230 FOR B=4 TO 6
2232 IF (A=6 OR A=11 OR A=16) AN
D B=4 THEN GOTO 2250
2235 IF 5*INT (A/5)=A AND B=4 TH
EN GOTO 2250
2240 IF A(A)=C AND A(A+B(B))=0 T
HEN GOTO 2270
2250 NEXT B
2260 NEXT A
2265 PRINT AT 20,0;"   EU PASSO
 "
2270 LET A(A+B(B))=C
2280 LET A(A)=0
2290 GOSUB 3000
2310 INPUT A$
2320 LET D=5*(CODE A$-38)+CODE A
$(2)-28
2330 LET E=5*(CODE A$(3)-38)+COD
E A$(4)-28
2340 LET A(E)=H
2350 LET A(D)=0
2360 GOTO 2165
3000 LET X=0
3010 LET Y=0
3012 PRINT AT 20,0;"██████████
██████"
3013 LET Z=RND**RND
3015 PRINT AT 20,0;"------------
--------"
3020 PRINT AT 5,0;"1 2 3 4 5"
3025 PRINT
3030 FOR A=1 TO 25
3040 IF A(A)=0 THEN PRINT "▒ ";
3045 IF A(A)<>0 THEN PRINT CHR$
(A(A));" ";
3050 IF 5*INT (A/5)=A THEN PRINT
 " ";CHR$ (A/5+37),,,;
3060 IF A(A)=C AND A>20 THEN LET
 X=X+1
3070 IF A(A)=H AND 5*INT (A/5)=A
 THEN LET Y=Y+1
3080 NEXT A
3090 PRINT "1 2 3 4 5"
3095 PRINT
3100 PRINT "HOMEM ";Y;" COMPUTAD
OR ";X
3101 IF Y>X THEN PRINT "VOCE EST
A GANHANDO"
3102 IF X>Y THEN PRINT "--EU EST
OU GANHANDO"
3105 IF X=4 OR Y=4 THEN GOTO 312
0
3110 RETURN
3120 GOTO 1120
5000 PRINT AT 6,9;"GRANDES JOGOS
"
5010 FOR U=0 TO 150
5020 NEXT U
5060 CLS
5070 PRINT AT 1,8;"CERCO ESTRATE
GICO"
5075 PRINT ,,,,"CADA JOGADOR TEM
 3 PEDRAS. O ZX"
5080 PRINT "JOGA COM AS ▣ E O HO
MEM COM AS ▣"
5085 PRINT "A JOGADA INICIAL PER
TENCE AO ZX"
5090 PRINT "E A FINALIDADE DO JO
GO CONSISTE"
5095 PRINT "EM CERCAR O ADVERSAR
IO IMPEDINDO"
6000 PRINT "ESTE DE ALCANCAR O B
ORDO OPOSTO"
6005 PRINT "DO TABULEIRO, AONDE
VAI PONTUAR."
6010 PRINT "AS PEDRAS MOVEM-SE E
M TODAS AS"
6015 PRINT "DIRECCOES, A DIREITO
 E EM DIAGO-"
```

```
6020 PRINT "NAL, EXCEPTO AS QUE
JA CHEGARAM"
6025 PRINT "A PONTUAR. O JOGO TE
M 2 VERSOES"
6030 PRINT "A *1* E A *2*, DEVE
INTRODUZIR O"
6035 PRINT "NUMERO DESEJADO."
6040 PRINT ,,"PRIMA **N/L** PARA
 COMECAR"
6045 INPUT L$
6050 CLS
6060 PRINT AT 7,15;"VERSAO 1"
6065 PRINT AT 10,15;"VERSAO 2"
6070 PRINT ,,,,"INTRODUZA O NUME
RO"
6099 RETURN
```

SONAR

ZX81 16 K

Autor: FERNANDO PRECES
Sacavém

```
   1 REM "6"
   8 DIM A(10,10)
  10 DIM B(10,10)
  13 GOSUB 1500
  15 GOSUB 400
  20 GOSUB 500
  25 GOSUB 600
  30 GOSUB 700
  32 GOSUB 1200
  35 GOSUB 800
  40 GOTO 20
 115 STOP
 400 LET V=30
 401 LET Z=5000
 404 FOR E=1 TO 6
 405 LET Y=INT (RND*10+1)
 410 LET X=INT (RND*10+1)
 415 LET A(X,Y)=1
 418 LET B(X,Y)=1
 420 NEXT E
 425 LET C=5
 430 LET D=5
 440 LET B(C,D)=2
 445 LET N=6
 499 RETURN
 500 PRINT AT 0,0;
 505 PRINT "  1234567890"
 510 FOR O=1 TO 10
 515 PRINT AT 0,13;".";AT 0,13;"
"
 520 IF O<10 THEN PRINT O;" ";
 522 IF O=10 THEN PRINT "0 ";
 524 FOR P=1 TO 10
 525 IF A(O,P)=1 AND B(O,P)=2 TH
EN LET N=N-1
 526 IF A(O,P)=1 AND B(O,P)<>1 T
HEN LET B(O,P)=4
 535 IF B(O,P)=0 THEN PRINT "▒";
 540 IF B(O,P)=1 THEN PRINT "█";
 545 IF B(O,P)=2 THEN PRINT "█";
 555 IF B(O,P)=4 THEN PRINT "+";
 560 NEXT P
 565 NEXT O
 567 LET V=V-1
 568 LET Z=Z-100
 569 IF V=0 THEN GOTO 1000
 570 PRINT AT 12,4;"SONAR"
 575 PRINT AT 0,20;" ▄█"
 580 PRINT AT 1,15;"------------
-----"
 585 PRINT AT 1,20;"▀▀▀▀▀▀▀"
 590 PRINT AT 5,21;"▒▒▒▒▒▒▒▒  (*";N
;")"
 598 IF N=0 THEN GOTO 1000
 599 RETURN
 600 FOR T=2 TO 11
 630 FOR U=2 TO 11
 635 PRINT AT T,U;"▒"
 660 NEXT U
 665 NEXT T
 699 RETURN
 700 FOR R=1 TO N
 735 LET H=INT (RND*10+1)
 740 LET I=INT (RND*10+1)
 745 IF B(H,I)=2 THEN GOTO 1100
 750 NEXT R
 795 RETURN
 800 PRINT AT 18,0;"DIRECCAO:(N
OU S)?"
 802 INPUT L$
 804 PRINT AT 18,0;"DIRECCAO:(E
OU O)?"
 806 INPUT M$
 808 PRINT AT 18,0;"
"
 810 LET B(C,D)=0
 825 IF L$="N" THEN LET C=C-1
 830 IF L$="S" THEN LET C=C+1
 835 IF M$="E" THEN LET D=D+1
 840 IF M$="O" THEN LET D=D-1
 845 LET B(C,D)=2
 850 RETURN
1000 PRINT AT 16,8;"PONTUACAO ";
Z
1010 STOP
1100 FOR O=1 TO 30
1110 PRINT AT 1,18;"TORPEDIADO"
1115 PRINT AT 1,18;"          "
1120 PRINT AT 1,18;"TORPEDIADO"
1130 NEXT O
1135 STOP
1200 IF B(X,Y)=4 THEN RETURN
1205 LET A(X,Y)=0
1210 LET B(X,Y)=0
1215 LET T=INT (RND*10+1)
1220 IF T=1 OR T=3 OR T=5 THEN L
ET X=X+1
1225 IF T=2 OR T=4 THEN LET X=X-
1
1230 IF T=6 OR T=8 OR T=10 THEN
LET Y=Y-1
1235 IF T=7 OR T=9 THEN LET Y=Y+
1
1240 IF X>10 THEN LET X=10
1245 IF Y>10 THEN LET Y=10
1246 IF X<1 THEN LET X=1
1247 IF Y<1 THEN LET Y=1
1250 LET A(X,Y)=1
1260 LET B(X,Y)=1
1270 RETURN
1500 PRINT "O SONAR E OS SUBMARI
NOS."
1510 PRINT ,,"VOCE COMANDA UM NA
VIO DE GUERRA,"
1520 PRINT ,,"TEM 6 SUBMARINOS N
O SEU RAIO DE"
1530 PRINT ,,"ACCAO. VOCE DESTRO
I UM SUBMARINO"
1540 PRINT ,,"QUANDO OCUPAR A SU
A CASA, MAS A"
1550 PRINT ,,"TODO O INSTANTE TA
MBEM PODE SER"
1560 PRINT ,,"TORPEDIADO. AO FIM
 RECEBE A PON-"
1570 PRINT ,,"TUACAO. BOA SORTE.
"
1580 PRINT ,,"PRIMA **N/L** PARA
 COMECAR."
1590 INPUT L$
1592 CLS
1595 RETURN
```

CARACTERES GRÁFICOS

ZX81

In. ZX COMPUTING, Dezembro/Janeiro 1984
Trad. e Adapt.: J. MAGALHÃES

O objectivo deste programa é tornar possível a criação de qualquer caracter de tamanho superior ao normal, através do seu armazenamento num «array». No final do programa, o «display» do écran é gravado na posição mais alta da RAM («high memory»), o que vai tornar possível a sua adaptação num outro programa.
A utilização do programa torna-se bastante interessante quer pelos seus efeitos quer pela capacidade de apresentação normal ou em inverse-vídeo.
O programa funciona igualmente com um número de caracteres inferior a 54 e torna possível a adaptção de novos caracteres com a opção de redefinir qualquer caracter.

COMO FUNCIONA O PROGRAMA

As linhas 20 a 240 referem-se à criação de um caracter gráfico e armazenamento de dados num «array». O «array» C é calculado pela subrotina na linha 640, havendo depois uma verificação da sua existência na linha 100. Quando a definição está completa, faça NEWLINE para introduzir a rotina PLOT que funciona como vamos descrever:
As linhas 260 e 270 indicam a posição inicial, a partir da qual é posicionado o primeiro caracter. A linha 290 chama um caracter. A subrotina na linha 640 converte C$ numa posição C no «array». Os «bytes» individuais armazenados nesta posição da «string» são então **carregados** («Poked») na primeira das 15 localizações da linha 1 (REM). Esta operação é levada a cabo pelas linhas 330 a 390.
A linha 410 copia o «display» do écran para o ponto mais alto da RAMTOP. A linha 420 coloca o caracter escolhido na posição inicial. As linhas 430 a 470 **estudam** o teclado para uma entrada («Input») com incrementação dos movimentos X-Y ou diminuição dos valores no endereço 16542 e 16544.
A linha 500 copia o écran no regresso da RAMTOP, e dá-se o salto para a linha 420 onde o caracter escolhido é re-apresentado. A linha 470 permite a saída deste «loop», facilitando a apresentação no écran do caracter escolhido, quando há nova execução da linha 410.
Quando a posição do caracter no écran está finalizada, fazendo NEWLINE, entra a rotina de inversão (INVERSE--VÍDEO) pela linha 560. A linha 570 armazena o «écran» na posição mais alta da RAMTOP. A opção final é para sair do programa pela tecla «S», ou NEWLINE para re-entrar na rotina PLOT.

INTRODUÇÃO DO PROGRAMA

Reserve parte da memória para as rotinas em Código Máquina, dando entrada directa dos seguintes comandos:

```
POKE 16388,0
POKE 16389,125
NEW
```

Isto coloca a RAMTOP em 32 000.
Dê entrada do programa «carregador» Hex. da fig. 1, assegurando-se que a linha 1 contém pelo menos 109 caracteres. Quando terminar esta fase, dê entrada do código Hex., como é apresentado na coluna da esquerda da fig. 2, terminando cada linha de código com NEWLINE.
Se ocorrer qualquer erro durante a introdução de dados, fazendo NEWLINE, seleccione «EDIT» (letra «E» em inverse vídeo). A partir deste momento pode fazer a correcção pela entrada correcta do código e continuar a introdução de dados.
Completa esta fase, dê entrada das letras «Z,Z». Apague todas as linhas excepto a linha 1; faça CLEAR. Dê entrada directa do comando DIM A$ (54, 5, 3) para criar um array de 54 caracteres.
Note que esta entrada não necessita de número de linha. Pode agora adicionar as linhas em BASIC apresentadas na fig. 3 (a partir da linha 20).
Depois de introduzida esta listagem, faça GOTO1. Use a listagem da fig. 4 para criar os caracteres desejados, dando entrada de um número seguido de NEWLINE por cada 15 entradas do caracter. Repare que o caracter vai sendo construído enquanto é dada cada entrada. Quando terminar todas as entradas, saia da rotina, fazendo NEWLINE. Pode ir passando as diferentes opções por sucessivas aplicações do comando NEWLINE. Accione a tecla «S» para saída do programa; se o quiser gravar, use GOTO 620.

COMO UTILIZAR O PROGRAMA

Assegure-se que a RAMTOP se encontra em 32 000. O programa entrará em funcionamento automaticamente logo que carregado.
Se por acaso for necessário iniciar o programa, faça GOTO 1 e, para responder à pergunta «QUALQUER CARACTER A CRIAR?», faça apenas NEWLINE.
Movimente o caracter até à posição desejada e fixe-o nesse ponto com as teclas previamente determinadas. O próximo caracter seleccionado aparecerá inicialmente sobreposto ao anterior, o que não significa erro pois pode deslocá-lo para a posição desejada.
A saída da rotina continua a ser NEWLINE. Se pretender os gráficos em Inverse-Vídeo, accione a tecla «S» quando for interrogado.
A figura 5 apresenta os caracteres já definidos. Para chamar os caracteres presentes no écran para um novo programa, faça NEWLINE e dê entrada do BASIC apresentado na fig. 6.
Faça RUN ao programa e, quando aparecer a mensaem 0/60, as linhas do programa podem ser apagadas.
Os gráficos são apresentados, quer pelo comando directo PRINT Z$, quer pelo comando numa linha de programa:

10 PRINT Z$, seguido de GOTO 10.

CÓDIGO MÁQUINA

As rotinas de Código Máquina tem os seguintes endereços:

16529	C.UP
16541	PRINT
16591	C.DN
16604	INVRT

As rotinas C.UP e C.DN usam a instrução LDIR para transferir o bloco de 727 bytes do endereço do «display file» para o endereço 32 000 e vice-versa.

As rotinas PRINT funcionam da seguinte forma: os registos A e B contêm X e Y (deslocamento do referencial) — écran 0,0. Os «steps» (espaços) 19 e 20 incrementam o endereço do «display file» (variável do sistema) de um valor igual ao que está no registo B.

Os «steps» 25 e 26 incrementam o endereço do «display file» de um valor igual a 33 vezes o valor do registo A.

Tendo estabelecido a posição do PRINT, os «steps» 31 a 34 fazem o PRINT dos primeiros 3 bytes dos caracteres aumentados.

Os «steps» 35 a 37 incrementam o endereço do «display file» em 30, indicando a posição correspondente ao PRINT da próxima linha, sendo aí apresentados os 3 bytes seguintes. Este processo é repetido 5 vezes para um caracter completo, de cada vez que o registo C, previamente carregado com cinco no «step» 27, tenha sido incrementado em 0 e regressa ao BASIC.

A rotina INVRT encontra cada caracter no «display file», acrescenta 128 ao valor presente e limpa o valor anterior.

LISTAGEM 1 — O programa de «carregamento» em Hex.

```
   1 REM :::::::::1:::::::::2:::
:::::3:::::::::4:::::::::5:::::
::::6:::::::::7:::::::::8::::::
::9:::::::::0:::::::::
  10 LET X=16514
  20 INPUT A$
  30 IF A$="" OR INT (LEN A$/2)<
>LEN A$/2 THEN GO TO 20
  40 PRINT AT 21,0;"          ";A
T 21,1;A$
  50 INPUT Z$
  60 IF INT (LEN Z$/2)<>LEN Z$(2
) THEN GO TO 50
  70 IF Z$<>"" THEN GO TO 100
  80 PRINT AT 21,0;"▒"
  90 GO TO 20
 100 SCROLL
 110 POKE X,16*CODE A$+CODE A$(2
)-476
 120 LET X=X+1
 130 LET A$=A$(3 TO )
 140 IF A$<>"" THEN GO TO 110
 150 IF Z$="ZZ" THEN STOP
 160 LET A$=Z$
 170 GO TO 40
```

LISTAGEM 2 — O código Hex. — introduza cada linha de código da coluna esquerda e faça NEWLINE.

```
000000                  1
000000                  2
000000                  3
000000                  4
000000                  5
2A0C40    LD HL,(16396) 6   C.UP
11007D    LD DE,32000   7
01D702    LD BC,727     8
EDB0      LDIR          9
C9        RET           10
0600      LD B,0        11  PRINT
3E00      LD A,0        12
F5        PUSH AF       13
112100    LD DE,33      14
2A0C40    LD HL,(16396) 15
78        LD A,B        16
FE00      CP 0          17
2803      JRZ,+3        18
23        INC HL        19
10FD      DJNZ,-3       20
F1        POP AF        21
FE00      CP 0          22
2804      JRZ,+4        23
47        LD B,A        24
19        ADD HL,DE     25
10FD      DJNZ,-3       26
0E05      LD C,5        27
118140    LD DE,16513   28
0603      LD B,3        29
13        INC DE        30
23        INC HL        31
1A        LD A,(DE)     32
77        LD (HL),A     33
10FA      DJNZ,-6       34
061E      LD B,30       35
23        INC HL        36
10FD      DJNZ,-3       37
0D        DEC C         38
20F0      JRNZ,-16      39
C9        RET           40
21007D    LD HL,32000   41  C.DN
ED5B0C40  LD DE,(16396) 42
01D702    LD BC,727     43
EDB0      LDIR          44
C9        RET           45
2A0C40    LD HL,(16396) 46  INVRT
0615      LD B,21       47
0E20      LD C,32       48
23        INC HL        49
7E        LD A,(HL)     50
C680      ADD A,128     51
77        LD (HL),A     52
0D        DEC C         53
20F8      JRNZ,-8       54
23        INC HL        55
10F3      DJNZ,-13      56
C9        RET           57
```

LISTAGEM 3 — A parte principal do programa em BASIC.

```
   1 REM                EERND) ?
▒NOT ▒ GOSUB ▒TAN ▒▒Y( PRINT )E
EERND? RETURN  C▒7( CLEAR LET RE
TURN  C▒ 7( CLEAR :▒)▒RND▒▒<7,"
 IF ▒27( CLEAR ▒4 LIST TAN 5 ? G
OSUB ?ERND▒NOT  ▒ GOSUB ▒TAN EERN
D▒+:47 SAVE 7( NEXT TAN
  20 REM ------ARRAY A$(54,5,3)
  30 REM -----------▒-----CRIAR
  40 PRINT AT 21,0;"CARACTER A C
RIAR ?"
  50 INPUT C$
  60 IF LEN C$>1 THEN GOTO 50
  70 CLS
  80 IF C$="" THEN GOTO 250
  90 GOSUB 640
 100 IF CODE A$(C,1)+CODE A$(C,2
)+CODE A$(C,3)+CODE A$(C,4)+CODE
 A$(C,5)=0 THEN GOTO 140
 110 PRINT AT 21,0;"RE-DEFINIR ?
(S)"
 120 INPUT B$
 130 IF B$<>"S" THEN GOTO 40
 140 CLS
 150 FOR M=1 TO 5
 160 PRINT """";C$;"""";" LINHA
";M;" DATA?   ";
 170 FOR N=1 TO 3
 180 INPUT D
 190 LET A$(C,M,N)=CHR$ D
 200 PRINT A$(C,M,N);
 210 NEXT N
 220 PRINT
 230 NEXT M
 240 GOTO 40
 250 REM ------▒----------PLOT
 260 LET X=0
 270 LET Y=16
 280 PRINT AT 21,0;"CARACTER PRE
TENDIDO ?"
 290 INPUT C$
 300 IF C$="" THEN GOTO 520
 310 PRINT AT 21,0;"PROCURO O CA
```

```
RACTER
 320 GOSUB 640
 330 LET Z=16514
 340 FOR M=1 TO 5
 350 FOR N=1 TO 3
 360 POKE Z,CODE A$(C,M,N)
 370 LET Z=Z+1
 380 NEXT N
 390 NEXT M
 400 PRINT AT 21,0;"MOVA 5,6,7,8
, C-CONFIRMAR"
 410 RAND USR 16529
 420 RAND USR 16541
 430 IF INKEY$="5" AND X>0 THEN
LET X=X-1
 440 IF INKEY$="8" AND X<29 THEN
 LET X=X+1
 450 IF INKEY$="7" AND Y>0 THEN
LET Y=Y-1
 460 IF INKEY$="6" AND Y<16 THEN
 LET Y=Y+1
 470 IF INKEY$="C" THEN GOTO 280

 480 POKE 16542,X
 490 POKE 16544,Y
 500 RAND USR 16591
 510 GOTO 420
 520 REM ---------------INVERTE
 530 PRINT AT 21,0;"INVERSE VIDE
O? (S)             "
 540 INPUT C$
 550 IF C$<>"S" THEN GOTO 570
 560 RAND USR 16604
 570 RAND USR 16529
 580 PRINT AT 21,0;"SAIDA DO PRO
GRAMA? (S)            "
 590 INPUT C$
 600 IF C$<>"S" THEN GOTO 280
 610 STOP
 620 SAVE "▒"
 630 GOTO 30
 640 LET C=CODE C$-10
 650 IF C=182 THEN LET C=1
 660 IF C=-10 THEN LET C=54
 670 RETURN
```

LISTAGEM 4 — Desta lista, escolha o(s) conjunto(s) de 15 entradas necessárias à criação do(s) seu(s) caractere(s). Depois faça NEWLINE.

```
" "    0   0   0   0   0   0   0
 0     0   0   0   0   0   0   0

""""   5   5   0   0   0   0   0
 0     0   0   0   0   0   0   0

"£"    135 3   4   129 4   0   129
 4     0   133 0   0   2   3   1

"$"    135 130 0   5   5   1   2
 7     1   134 130 1   0   1   0

":"    0   0   0   0   0   0   1
 0     0   0   0   0   1   0   0

"?"    6   3   4   1   0   5   0
 7     0   0   1   0   0   1   0

"("    135 1   0   5   0   0   5
 0     0   134 0   0   0   1   0

")"    134 0   0   0   5   0   0
 5     0   135 1   0   1   0   0

">"    0   0   0   134 0   0   135
 1     0   1   0   0   0   0   0

"<"    0   0   0   135 1   0   134
 0     0   0   1   0   0   0   0

"="    0   0   0   131 131 4   131
 131   4   0   0   0   0   0   0

"+"    0   0   0   0   5   0   3
 7     1   0   1   0   0   0   0

"-"    0   0   0   0   0   0   3
 3     1   0   0   0   0   0   0

"*"    0   4   0   134 130 1   132
 128   1   1   5   1   0   0   0

"/"    0   0   5   0   133 0   135
 1     0   6   0   0   1   0   0

";"    0   0   0   0   0   0   2
 0     0   0   0   0   6   0   0

","    0   0   0   0   0   0   0
 0     0   0   0   0   6   0   0

"."    0   0   0   0   0   0   0
 0     0   0   0   0   1   0   0

"0"    6   3   4   5   135 5   130
 1     5   5   0   5   2   3   0

"1"    132 0   0   133 0   0   133
 0     0   133 0   0   3   1   0

"2"    6   3   4   0   135 1   135
 1     0   5   0   0   3   3   1

"3"    3   3   5   0   6   0   2
 3     4   4   0   5   2   3   0

"4"    0   6   0   6   0   0   5
 133   0   3   132 1   0   2   0

"5"    6   3   0   5   0   0   3
 3     4   4   0   5   2   3   0

"6"    6   3   4   5   0   0   7
 3     4   5   0   5   2   3   0

"7"    3   3   5   0   135 1   135
 1     0   133 0   0   2   0   0

"8"    6   3   4   5   0   5   6
 3     4   5   0   5   2   3   0

"9"    6   3   4   5   0   5   2
 3     5   4   0   5   2   3   0

"A"    6   3   4   5   0   5   7
 3     5   5   0   5   1   0   1

"B"    7   3   4   5   0   5   7
 3     4   5   0   5   3   3   0

"C"    6   3   4   5   0   0   5
 0     0   5   0   4   2   3   0

"D"    7   3   4   5   0   5   5
 0     5   5   0   5   3   3   0

"E"    7   3   1   5   0   0   7
 3     1   5   0   0   3   3   1

"F"    7   3   1   5   0   0   7
 3     1   5   0   0   1   0   0

"G"    6   3   4   5   0   0   5
 2     5   5   0   5   2   3   0

"H"    5   0   5   5   0   5   7
 3     5   5   0   5   1   0   1

"I"    132 1   0   133 0   0   133
 0     0   133 0   0   3   1   0

"J"    0   0   5   0   0   5   0
 0     5   5   0   5   2   3   0
```

```
"K"   5   0   5   5   6   0   7
 4    0   5   2   4   1   0   1

"L"   5   0   0   5   0   0   5
 0    0   5   0   0   3   3   1

"M"   7   6   5   5   5   5   5
 0    5   5   0   5   1   0   1

"N"   5   0   5   7   4   5   5
 2    5   5   0   5   1   0   1

"O"   6   3   4   5   0  5   5
 0    5   5   0   5   2   3   0

"P"   7   3   4   5   0   5   7
 3    0   5   0   0   1   0   0

"Q"   6   3   4   5   0   5   5
 0    5   5  134  1   2   1   1

"R"   7   3   4   5   0   5   7
 7    0   5   2   4   1   0   1

"S"   6   3   4   5   0   0   2
 3    4   4   0   5   2   3   0

"T"   3   7   1   0   5   0   0
 5    0   0   5   0   0   1   0

"U"   5   0   5   5   0   5   5
 0    5   5   0   5   2   3   0

"V"   5   0   5   5   0   5   5
 0    5  134 135  1   0   1   0

"W"   5   0   5   5   0   5   5
 4    5   5   5   5   3   2   1

"X"   5   0   5   2   6   0   0
 5    0   6   2   4   1   0   1

"Y"   5   0   5  134 135  1   0
 5    0   0   5   0   0   1   0

"Z"   3   3   5   0  135  1  135
 1    0   5   0   0   3   3   1
```

FIGURA 5 — Os caracteres criados pelo programa.

1234567890
ABCDEFGHIJK
LMNOPQRSTU
VWXYZ*£($/?

LISTAGEM 6 — Utilize este programa para introduzir em qualquer programa (BASIC) os caracteres já definidos.

```
  10 FAST
  20 LET Z$=""
  30 FOR X=32000 TO 32692
  40 IF PEEK X<>118 THEN LET Z$=
Z$+CHR$ PEEK X
  50 NEXT X
  60 SLOW
```

ANIMAÇÃO NO SPECTRUM

16 e 48 K

In. YOUR COMPUTER, Abril 1983
Trad. e Adapt.: J. MAGALHÃES

A possibilidade de cor, som e capacidade de gráficos de alta redução no SPECTRUM encorajou muitas pessoas a escrever programas de jogos. É claro que a maioria deles exige bastante velocidade para maior entusiamo.

O método mais utilizado para produzir ilusão de movimento começa por uma posição inicial do objecto, seguindo-se uma nova posição mais à frente e o desaparecimento da anterior. Por vezes este método é suficiente para um programa em BASIC, com o inconveniente de se aperceber um tipo de «empurrão» à figura inicial.

Dentro do software comercial, utiliza-se o código máquina para se obter um movimento mais real e mais rápido. Ora, isto não está acessível a muitas pessoas que programem para si mesmas em suas casas. Será necessário considerar outros métodos de se conseguir a veiocidade em programas BASIC.

Por isso, vamos apresentar uma pequena rotina em código máquina que, esperamos, satisfaça as suas necessidades. Esta rotina foi aperfeiçoada de modo a poder utilizar gráficos («user-defined») de qualquer tamanho, sendo chamados sempre que pretendidos. Podem-se deslocar em qualquer posição, num écran de alta resolução 256x176, o que já é considerável em relação ao écran 32x22 (o écran normal).

O código máquina ocupa 197 bytes que podem ser guardados deslocando a RAMTOP na zona mais abaixo da memória, usando para isso o comando CLEAR e colocando o código entre a área BASIC e a área de Gráficos.

NOTA: Os endereços diferem no 48K e no 16K sendo, no entanto, a mesma rotina. O bloco de bytes do código máquina encontra-se na listagem 1.

Os possuidores do SPECTRUM 48K podem usar o programa de carregamento da listagem 2. Dê entrada deste programa e faça RUN; introduza os bytes um a um da esquerda para a direita. Antes de fazer a gravação (depois de ter dado todas as entradas), sugerimos que confira os dados retirando do programa (listagem 2) as linhas 20, 30, 70 e 90 e substituindo a linha 80 por:

80 PRINT TAB 10; PEEK i

Se já conferiu, então grave da seguinte forma:

SAVE «animate» CODE 65160, 197

No caso do SPECTRUM 16K, serão necessárias as seguintes alterações na listagem 2:

— Linha 20 será CLEAR 32389
— Linha 50 será FOR i = 32390 TO 32586.

Para gravar:

SAVE «animate» CODE 32390, 197

Agora pode testar a rotina, limpando o écran (CLS; ENTER) e usando o comando:

RANDOMIZE USR 65171 — SPECTRUM 48K
RANDOMIZE USR 32401 — SPECTRUM 16K

No canto superior esquerdo do écran deverá aparecer o caracter «A». Se isso não acontecer é porque houve qualquer engano na introdução do código máquina.
Se o programa «desaparecer», recarregue-o (previamente gravado) sem esquecer limpar a RAMTOP pelo CLEAR apropriado. Verifique.
Se tudo correu tal como indicámos, passa-se ao programa BASIC.
Repare na listagem 3 (rotina «disassembled») que os primeiros 6 bytes são usados como variáveis XPOS, YPOS, etc..
Estes nomes não serão reconhecidos se tentar usá-los no programa BASIC; mas, fazendo diferentes valores para o POKE nestes 6 bytes, pode controlar a operação da rotina como se segue.

CONTROLO DA ROTINA

- XPOS — endereço 32390 para o SPECTRUM 16K
  — endereço 65160 para o SPECTRUM 48K

«X» é a coordenada da posição no canto superior esquerdo do écran (posição inicial do gráfico). São possíveis outros valores como para o comando PLOT de 0 a 255.

- YPOS — endereço 32391 (SPECTRUM 16K)
  — endereço 65161 (SPECTRUM 48K)

«Y» é a coordenada da posição inicial do gráfico, no canto superior esquerdo do écran. Pode tomar valores de 0 a 175 (topo do écran).

- EXTENSÃO («width») — endereço 32392 (16K)
  — endereço 65162 (48K)

Representa o número de pontos da esquerda para a direita.

- HT — endereço 32393 (16 K)
  — endereço 65163 (48 K)

Número de pontos do fundo ao topo do écran.

- MODE — endereço 32394 (16 K)
  — endereço 65164 (48 K)

Toma os valores 1 se deseja o caracter na presente posição, ou 0 se deseja limpar um bloco de extensão × pontos HT na presente posição.

- UDGCH — endereço 32395 (16 K)
  — endereço 65165 (48 K)

Deve tomar valores entre 1 a 21, que transmite à rotina onde encontrar os elementos DATA do gráfico que deseja reproduzir. O valor 1 significa que o 1.° byte de DATA se encontra no endereço USR «a», que é o 1.° byte do gráfico «a». O valor 21 significa que se encontra no endereço USR «u».
Os valores POKE nestes 6 bytes, antes de chamada a rotina, serão alterados à saída. Por isso, se utilizar mais de um caracter não precisa renovar («update») todos os bytes.
Terá resultados bastante estranhos ou pode até perder o programa se pretende valores para XPOS e YPOS que não permitam a posição no espaço que compreende o écran. Então o seu programa BASIC precisa prever este caso.

DIFERENTES CORES

A rotina apresenta o caracter gráfico com a cor definida pelos dados do sistema variável ATTR T — localização 23695. Se pretender cores diferentes pode usar ATTR T a partir do BASIC, sem alterar a presente posição, por uma simulação da instrução PRINT:

PRINT PAPER 6; INK 1;

Antes de ser chamada a rotina. Assim, podem ser usados os comandos FLASH e BRIGHT.
Se usar a rotina para gráficos de 8 por 8 pontos («user-graphics»), o método de armazenamento é exactamente igual ao descrito no capítulo 14 do manual do SPECTRUM. Contudo, a rotina pode produzir gráficos de qualquer dimensão, mas através de um método diferente.
Inicialmente, desenhe o caracter numa folha de papel quadriculado. Dado que vamos usar a rotina para mover o caracter de 2 em 2 pontos, é aconselhável deixar uma margem de duas colunas vazias à esquerda e à direita. A operação que faz mover o caracter da esquerda para a direita em 2 pontos, apresentando uma nova posição, irá automaticamente cobrir o anterior «invasor» (o 1.° caracter), evitando erros. O caracter que queremos usar é de 14 por 11 pontos.
Se a extensão não for rigorosamente múltipla de 8, deverá acrescentar o número de colunas vazias necessárias do lado direito. A extensão do caracter é então dividida em oito pequenas secções que podem ser descritas pelo vulgar número BIN e armazenadas na área de gráficos «user-defined»).
Para isso, pode usar-se um programa como o da listagem 4.
O 1.° byte é armazenado no endereço dado por USR «a».
Aquando da entrada de dados para os seus caracteres, lembre-se que DATA para a 1.ª linha é armazenado em primeiro lugar, depois a 2.ª linha, e assim sucessivamente.
Desde que o «invasor» ocupe 22 bytes (isto é, todos os gráficos «a», «b» e a maior parte de «c»), a secção seguinte não usada começará em USR «d», onde podem ser armazenados os 4 bytes de DATA necessários para o caracter da figura 2.
No invasor, as pernas aparecerão em movimento enquanto ele se move no écran. Dê entrada e faça RUN da listagem 4.
Se quiser, grave estes gráficos:

SAVE «chars» CODE USR «a», 32

Pode limpar o écran com NEW, desde que a rotina de gráficos se encontre na parte superior da RAMTOP — e introduza o programa da listagem 5. Aparecerá uma linha de 7 invasores multicolores, movimentando-se tal como no tradicional jogo «INVADERS».

NOTA: Os invasores podem ser apagados se for chamada a rotina quando eles se movem uma linha abaixo, bastando para isso alterar **Mode** para 0 (zero), **Extensão** para 140 e **HT** para 11 (linhas 130-160).

OUTRAS SECÇÕES DO PROGRAMA (listagem 5)

- **Linhas 300 - 340**
  — produzem a linha dos 7 caracteres com cores (INK) diferentes.
- **Linhas 200 - 260**
  — chamam a subrotina na linha 300, para apresentar primeiro a linha dos invasores em determinada posição, acrescentando seguidamente as pernas com o caracter da figura 2.
- **Linhas 100 - 170**
  — fazem o movimento inicial esquerda-direita no ecran, e depois movimento inverso.
- **Linhas 20 - 60**
  — originam a descida dos invasores uma linha, depois de terem percorrido a anterior.

AUMENTO DE VELOCIDADE

A partir deste exemplo pode verificar a vantagem da utilização da rotina apresentada num seu programa BASIC.
No final, verificará que a área de "user-defined graphics" não é suficiente para armazenar a quantidade ou o tamanho que pretende para os caracteres; pode recolocar a rotina e usar uma área maior da RAM para armazenamento dos caracteres.
Isso consegue-se facilmente, desde que a rotina em código-máquina possa usar-se em qualquer parte, contando ainda que o 1.º byte (isto é, XPOS) venha imediatamente a seguir a RAMTOP. Por exemplo, se desejasse a rotina para gerar 40 caracteres por linha, precisaria definir o seu caracter em 6 por 8, deixando livres as margens esquerda e direita, como já foi dito. Isto exigiria uma grande área de memória — cerca de 800 bytes — para armazenamento. Portanto, os utilizadores do SPECTRUM 48 K teriam de localizar a rotina no endereço 64000, seguindo o comando CLEAR 63999 e carregando em seguida:

LOAD "animate" CODE 64000

A área de gráficos iniciará agora no endereço 64300, fazendo o POKE ao sistema de variáveis UDG — localização 23675/6. Isto dá um amplo espaço para um caracter alternativo ou para quaisquer outros gráficos.

NOTA IMPORTANTE — Listagem 6

Devido a um erro na rotina do código-máquina, que mais tarde e veio a verificar pelo facto de não funcionar quando **Width** maior que 16 bits, deve ser adicionado o programa BASIC da listagem 6.
A razão deste erro deve-se à falta da instrução LD (IX + 9), 8 que deveria aparecer entre as instruções EX HL, DE e L4 DEC (IX + 8) — Listagem 3.
Pegue na primeira versão da rotina em código-máquina "animate" e, com o apropriado comando CLEAR, passe a RAMTOP para 65158 (no caso do SPECTRUM 48 K) ou 32389 (no caso de 16 K). Dê entrada e faça RUN do programa apresentado na listagem 6.
Os utilizadores do SPECTRUM 16 K, deverão alterar a linha 10 para:

10 LET a = 32390

Listagem 1 — Código-máquina

2	175	8	8	1	1
0	0	0	0	0	221
42	178	92	221	35	42
123	92	221	94	5	29
203	35	203	35	203	35
22	0	25	235	221	78
0	221	70	1	221	126
3	245	197	221	126	2
221	119	6	205	170	34
221	119	7	221	52	7
47	230	7	60	221	119
8	213	229	205	219	11
225	209	221	203	4	70
32	5	1	0	0	24
5	235	70	35	78	235
221	54	9	8	221	54
10	9	126	221	53	7
40	9	7	221	53	10
221	53	7	32	247	203
33	203	16	23	221	53
10	221	53	6	32	19
221	53	10	40	6	7
221	53	10	32	250	119
193	241	5	61	32	155
201	221	53	9	32	10
221	203	4	70	40	4
235	35	78	235	221	53
8	32	204	221	53	10
40	6	7	221	53	10
32	250	119	35	213	229
245	205	219	11	241	225
209	221	54	8	8	221
54	7	1	24	153	

Listagem 2 — Para o SPECTRUM 16 K, alterar as linhas 20 e 50:
20 CLEAR 32389
50 FOR I = 32390 TO 32586

```
 10 REM listing 2 (48K)
 20 CLEAR 65159
 30 CLS : PRINT "Entrada dos nu
meros - listagem 1"
 40 PRINT : PRINT "Endereco"
 50 FOR i=65160 TO 65356
 60 PRINT i
 70 INPUT n
 80 PRINT TAB 10;n
 90 POKE i,n
100 NEXT i
```

Listagem 3 — Código-Máquina "disassembled"

```
        ORG   65160
XPOS    DEFB  2
YPOS    DEFB  175
WIDTH   DEFB  8
HT      DEFB  8
MODE    DEFB  1
UDGCH   DEFB  1
        DEFB  0
        DEFB  0
        DEFB  0
        DEFB  0
        DEFB  0
;
START   LD    IX,(23730)
        INC   IX
        LD    HL,(23675)
        LD    E,(IX+5)
        DEC   E
        SLA   E
        SLA   E
        SLA   E
        LD    D,0
        ADD   HL,DE
        EX    HL,DE
        LD    C,(IX+0)
        LD    B,(IX+1)
        LD    A,(IX+3)
LINE    PUSH  AF
        PUSH  BC
        LD    A,(IX+2)
        LD    (IX+6),A
        CALL  22AAH
        LD    (IX+7),A
        INC   (IX+7)
        CPL
        AND   7
        INC   A
        LD    (IX+8),A
        PUSH  DE
        PUSH  HL
        CALL  0BDBH
        POP   HL
        POP   DE
        BIT   0,(IX+4)
        JR    NZ,5
        LD    BC,0
        JR    LN1
        EX    HL,DE
        LD    B,(HL)
        INC   HL
        LD    C,(HL)
        EX    HL,DE
LN1     LD    (IX+9),8
L1      LD    (IX+10),9
        LD    A,(HL)
        DEC   (IX+7)
        JR    Z,L3
L2      RLCA
        DEC   (IX+10)
        DEC   (IX+7)
        JR    NZ,L2
L3      SLA   C
        RL    B
        RLA
        DEC   (IX+10)
        DEC   (IX+6)
        JR    NZ,L7
        DEC   (IX+10)
        JR    Z,L6
L5      RLCA
        DEC   (IX+10)
        JR    NZ,L5
L6      LD    (HL),A
        POP   BC
        POP   AF
        DEC   B
        DEC   A
        JR    NZ,LINE
        RET
L7      DEC   (IX+9)
        JR    NZ,L4
        BIT   0,(IX+4)
        JR    Z,L4
        EX    HL,DE
        INC   HL
        LD    C,(HL)
        EX    HL,DE
L4      DEC   (IX+8)
        JR    NZ,L3
        DEC   (IX+10)
        JR    Z,L8
L9      RLCA
        DEC   (IX+10)
        JR    NZ,L9
        JR    NZ,L3
L8      LD    (HL),A
        INC   HL
        PUSH  DE
        PUSH  HL
        PUSH  AF
        CALL  0BDBH
        POP   AF
        POP   HL
        POP   DE
        LD    (IX+8),8
        LD    (IX+7),1
        JR    L1
        END
```

Listagem 4

```
  10 REM chrs fig.1
  20 FOR i=USR "a" TO USR "a"-21
  30 READ n: POKE i,n
  40 NEXT i
  50 DATA BIN 00000111,BIN 10000
000,BIN 00011111,BIN 11100000,BI
N 00111111,BIN 11110000,BIN 0011
0011,BIN 00110000
  60 DATA BIN 00110011,BIN 00110
000,BIN 00111111,BIN 11110000,BI
N 00011111,BIN 11100000,BIN 0001
0100,BIN 10100000
  70 DATA BIN 00010011,BIN 00100
000,BIN 00100000,BIN 00010000,BI
N 00100000,BIN 00010000
  80 REM chrs fig.2
  90 FOR i=USR "d" TO USR "d"+3
 100 READ n: POKE i,n
 110 NEXT i
 120 DATA BIN 00001000,BIN 01000
000,BIN 00001000,BIN 01000000
```

Listagem 5

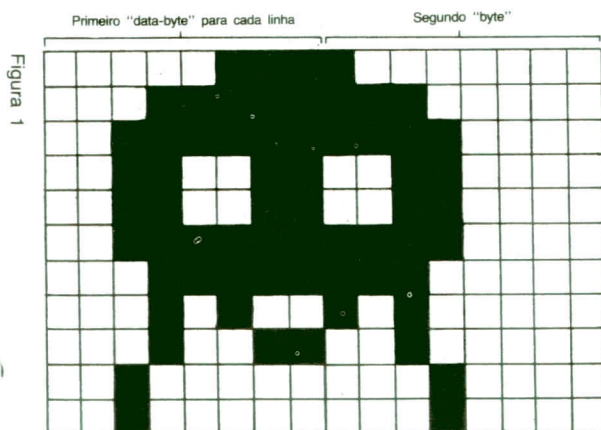
```
   4 REM ----------------------
   5 CLEAR 65159
  10 PAPER 7: BORDER 7: CLS
  20 FOR y=160 TO 100 STEP -20
  30 GO SUB 100
  40 NEXT y
  50 INK 0
  60 STOP
  70 REM ***********************
 100 POKE 65162,14: POKE 65164,1
 110 FOR x=0 TO 100 STEP 2: GO S
UB 200: NEXT x
 120 FOR x=100 TO 0 STEP -2: GO
SUB 200: NEXT x
 130 POKE 65160,0: POKE 65161,y
 140 POKE 65162,140: POKE 65163,
11
 150 POKE 65164,0
 160 RANDOMIZE USR 65171
 170 RETURN
 180 REM ***********************
 200 POKE 65161,y: POKE 65163,11
 210 POKE 65165,1
 220 GO SUB 300
 230 POKE 65161,y-9: POKE 65163,
2
 240 POKE 65165,4
 250 GO SUB 300
 260 RETURN
 270 REM ***********************
 300 FOR c=0 TO 6: PRINT INK c;
 310 POKE 65160,c*20+x
 320 RANDOMIZE USR 65171
 330 NEXT c
 340 RETURN
 350 REM ***********************
```

Listagem 6 — Para o SPECTRUM 16 K, alterar a Linha 10:
10 LET a = 32390

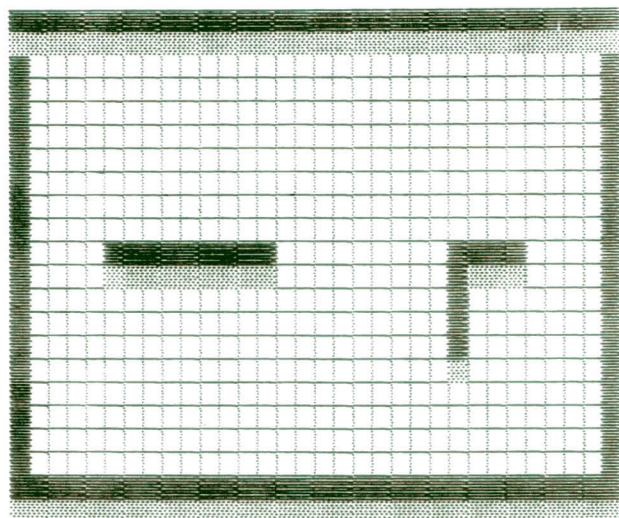
```
  10 LET a=65160
  20 POKE a+158,24
```

```
30 POKE a+159,37
40 FOR b=a+197 TO a+204
50 READ c: POKE b,c
60 NEXT b
70 DATA 78,235,221,54,9,8,24,2
11
80 SAVE "animate"CODE a,205
```

Figura 1

Figura 2

PERSEGUIÇÃO — SPECTRUM 16 K

In. HOME COMPUTING
Trad. e Adapt.: ANTÓNIO AMARAL/Porto

```
 O objectivo da PERSEGUCAO e
empurrar o outro jogador para
uma parede.

 Cada jogador deixa uma parede
e nao pode parar ate que um seja
destruido.

              CONTROLES

               ^ < > v

Jogador 1.     2 0 W A

Jogador 2:     0 O P L

               ^ < > v
```

```
   1 REM PERSEGUICAO
   2 GO SUB 9000
   3 GO SUB 8000
   5 LET s1=0: LET s2=0
  10 PRINT AT 0,0;"████████████████████████████████"
  20 PRINT AT 20,0;"████████████████████████████████"
  30 PRINT AT 21,0;"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
  40 PRINT AT 1,0;"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
  50 FOR f=2 TO 19: PRINT AT f,0
;"█UUUUUUUUUUUUUUUUUUUUUUUUUUUUUU█"
  60 NEXT f
  70 LET x=4: LET y=10
  80 LET a=27: LET b=10
  90 LET d=2: LET c=4
  95 GO TO 100+60*(RND<.5)
 110 LET a$=("1" AND IN 63486<>2
55)+("2" AND IN 64510=253)+("3"
AND IN 65022<>255)+("4" AND IN 6
4510=254)
 120 IF LEN a$=1 THEN LET d=VAL
a$
 125 LET x=x+(d=2)-(d=4): LET y=
y+(d=3)-(d=1)
 126 IF SCREEN$ (y,x)<>"" THEN G
O TO 2000
 130 PRINT AT y,x; BRIGHT 1; INK
 c1;"█"
 140 IF SCREEN$ (y+1,x)="" THEN
PRINT AT y+1,x; PAPER c1; INK c1
;"▒"
 145 BEEP .01,0
 160 LET a$=("1" AND IN 61438<>2
55)+("2" AND IN 57342=254)+("3"
AND IN 49150<>255)+("4" AND IN 5
7342=253)
 170 IF LEN a$=1 THEN LET c=VAL
a$
 175 LET a=a+(c=2)-(c=4): LET b=
b+(c=3)-(c=1)
 176 IF SCREEN$ (b,a)<>"" THEN G
O TO 1000
 180 PRINT AT b,a; BRIGHT 1; INK
 c2;"█"
 190 IF SCREEN$ (b+1,a)="" THEN
PRINT AT b+1,a; PAPER c2; INK c2
;"▒"
 195 BEEP .01,10
 200 GO TO 100
1000 PRINT AT b,a; FLASH 1;" "
1010 LET s1=s1+1
1020 GO TO 3000
2000 PRINT AT y,x; FLASH 1;" "
2010 LET s2=s2+1
3000 FOR f=30 TO 60: BEEP .001,f
: NEXT f
3010 PRINT INVERSE 1;AT 9,11;"Jo
gador 1: ";s1
3020 PRINT INVERSE 1;AT 11,11;"J
ogador 2: ";s2
3030 FOR f=60 TO 30 STEP -1: BEE
P .001,f: NEXT f
3035 IF s1>9 OR s2>9 THEN GO TO
4000
3040 IF INKEY$<>"" THEN GO TO 30
40
3050 PRINT AT 20,3; INVERSE 1;"C
ARREGUE EM QUALQUER TECLA"
3060 IF INKEY$<>"" THEN GO TO 10
3070 GO TO 3060
4000 PRINT FLASH 1;AT 13,10;"Jog
ador ";(s2=10)+1;" GANHA"
4010 INPUT "Outro jogo? "; LINE
a$
4020 IF a$="" THEN GO TO 4010
4030 IF a$(1)<>"n" THEN RUN
4040 STOP
```

```
8000 BORDER 0: PAPER 0: INK 7: C
LS
8010 PRINT "Este e a :"
8020 PRINT TAB 13;"PERSEGUICAO"
8030 PRINT '" O objectivo da PER
SEGUCAO e      empurrar o outro jo
gador para    uma parede."
8040 PRINT '" Cada jogador deixa
 uma parede   e nao pode parar at
e que um sejadestruido."
8050 PRINT 'TAB 12;"CONTROLES"
8055 PRINT 'TAB 14;"^ < > v"
8060 PRINT '"Jogador 1.     2 0 W
A"
8070 PRINT '"Jogador 2:     0 O P
L"
8080 PRINT 'TAB 14;"^ < > v"
8100 INPUT "Cor do jagador 1? (1
 A 7) ";c1
8110 IF c1<1 OR c1>7 THEN GO TO
8100
8120 INPUT "Cor do jogador 2? (1
 A 7) ";c2
8130 IF c2<1 OR c2>7 THEN GO TO
8120
8140 RETURN
9000 FOR f=0 TO 7 STEP 2: POKE U
SR "a"+f,170: POKE USR "a"+f+1,8
5: NEXT f
9010 FOR f=0 TO 6: POKE USR "b"+
f,128: NEXT f: POKE USR "b"+7,25
5
9020 RESTORE : FOR f=0 TO 7: REA
D a: POKE USR "c"+f,a: POKE USR
"d"+7-f,a: NEXT f
9030 DATA 0,0,8,20,34,0,0,0
9040 RETURN
```

```
GRAFICOS:

Linha 50           - G + teclas 8/B
Linhas 140, 190    - G + tecla A
Linhas 8055,8080-  G + teclas C,D
```

ROTINAS DE APLICAÇÃO NO ZX SPECTRUM

«Acontece muitas vezes que, ao corrigir ou alterar um programa, este não tem espaço entre linhas sucessivas ou, se tem, não fica tão compreensível com as novas linhas a adicionar. Isto torna a tarefa do programador bastante demorada e cansativa.» (Hugo Assumpção/Lisboa).

ROTINA 1 — «REORD - 9991»

HUGO ASSUMPÇÃO

Serve para reordenar linhas, com o «salto de linha» desejado: geralmente de 5 em 5 ou de 10 em 10. Tem a limitação de ter de se alterar os GO TOs e GO SUBs pontualmente, isto é, na própria linha.

```
9990 STOP
9991 LET b=PEEK 23635+256*PEEK 2
3636
9992 INPUT "Salto das linhas",a
 LET n=a: IF a>100 THEN GO TO 99
92
9993 IF 256*PEEK b+PEEK (b+1)=99
90 THEN STOP
9994 POKE b,INT (n/256)
9995 POKE b+1,n-256*INT (n/256)
9996 LET n=n+a
9997 LET b=b+4+PEEK (b+2)+256*PE
EK (b+3)
9998 GO TO 9993
```

NOTAS:

Linha 9990 — Impede que o programa principal siga para este.
» 9991 — Calcula o endereço do início do programa.
» 9992 — Introdução dos saltos de linha.
» 9993 — Verifica se a linha a alterar é a primeira deste subprograma e pára se fôr.
» 9994 a 9996 — Corrige e actualiza a linha do programa.
» 9997 — Salta para a linha seguinte a corrigir.
» 9998 — Retoma o ciclo

ROTINA 2 — «ERASE LINE GOTO 9991»

HUGO ASSUMPÇÃO

Permite apagar blocos de linhas de um programa. É também útil para quem trabalha com subrotinas standard, isto é, começando e acabando em determinada linha do programa.

```
9990 STOP
9991 LET x=PEEK 23635+256*PEEK 2
3636: INPUT "1a Linha",a: INPUT
"ultima linha",b: IF b<a OR b>=9
990 THEN GO TO 9991
9992 LET cl=PEEK (x+2)+256*PEEK
(x+3): IF PEEK x*256+PEEK (x+1)=
a THEN LET i=x
9993 IF PEEK x*256+PEEK (x+1)=99
90 THEN STOP
9995 IF PEEK x*256+PEEK (x+1)=b
THEN LET f=x+cl+4: GO TO 9999
9997 LET x=x+cl+4: GO TO 9992
9999 LET c=f-i-4: POKE i+2,c-256
*INT (c/256): POKE i+3,INT (c/25
6): CLS : PRINT INVERSE 1;"apagu
e a linha";a: LIST a
```

NOTA:

Linha 9991 — Calcula o endereço do início do programa principal e dá entrada das primeira (a) e última (b) linhas.
» 9992 — cL significa o Comprimento da Linha a examinar. Verifica se está perante a 1.ª linha a apagar(a) e, se sim, introduz o seu endereço em «i».
» 9993 — Verifica se a linha presente é 9990. Se sim, pára — de outra forma apagaria o subprograma.
» 9995 — Verifica se está perante a última linha a apagar (b) e, se sim, coloca o seu endereço em «f».
» 9997 — Salta para o endereço da linha seguinte e retoma o ciclo.
» 9999 — Calcula a diferença entre o 1.º endereço de «a» e o último de «b». Guarda-o em «c».
Atribui à primeira linha a alterar, «a», comprimento igual a «c».
Pede para apagar a linha «a».

IMPORTANTE — A primeira e a última linha do programa têm que existir sempre.

Para aplicação dos rotinas 1 e 2, faça o MERGE do programa em que pretende a sua utilização.
Tenha em atenção que esse seu programa não ocupe as mesmas linhas da rotina.

ROTINA 3

In. POPULAR COMPUTING, N.° 24
Adapt. e Trad.: J. MAGALHÃES

Torna mais fácil o uso da instrução DELETE, através da tecla «Symbol Shift».

Como concerteza já se apercebeu, para usar a tecla DELETE (apagar qualquer instrução), tem obrigatoriamente de usar as teclas CAPS SHIFT + DELETE. Ora, talvez não seja esta a forma mais prática pois exige a ocupação das duas mãos. Assim, pensamos substituir a tecla CAPS SHIFT pela tecla SYMBOL SHIFT que fica mais próxima da DELETE, permitindo um manejamento mais prático (SYMBOL SHIFT e DELETE).

Esta pequena rotina em Código Máquina pode ser adaptada para o SPECTRUM 16 K se se alterar a linha 10 para:

10 FOR g = 32339 TO 32377 : READ a : POKE g, a : NEXT g

(No SPECTRUM 48 K, a listagem não sofre qualquer alteração).

Para iniciar a rotina, faça:

PRINT USR 65120

```
  5 CLEAR 65120
 10 FOR g=65120 TO 65158: READ
a: POKE g,a: NEXT g
 15 DATA 62,9,237,71,237,94,201
,0,0,255,243,245,229,33,8,92,126
,254,95,40,8,254,12,40,9,225,241
,251,201,62,12,119,24,247,62,95,
119,24,242
```

CARACTERES GRÁFICOS

SPECTRUM

In. POPULAR COMPUTING, 51
Trad. e Adapt.: J. MAGALHÃES

Estas duas rotinas produzem caracteres com tamanho superior ao normal, tal como pode verificar pela Fig. 1.

```
0 1 2 3 4 5 6 7 8 9
A B C D E F G H I J
K L M N O P Q R S T
U V W X Y Z   .   ,
*   /   (   )   =
+   ↑   <   >   ?
£   !   =   ;   :
```

Depois de ter introduzido a listagem 1, faça RUN e prepare o gravador para gravação do «array» numérico quando aparecer no écran:

«GRAVAR ARRAY NUMERICO»
«Start tape, then press any Key»

Depois de gravado, pode automaticamente verificar, bastando trocar os cabos do gravador e do Spectrum para o EAR. Dê entrada da rotina da listagem 2 e grave-a à parte. Pode fazer um «MERGE» desta rotina em qualquer programa onde queira que apareçam estes caracteres; mas esse programa deve conter a seguinte linha:

LOAD «LARGECHARS» DATA t()

NOTAS: — Cada palavra não pode ter mais que 10 caracteres.
— Pode usar-se minúsculas, mas a apresentação será sempre em maiúsculas (Fig. 1).
— As palavras a alterar devem ser carregadas numa «string» (Z$), antes de fazer o MERGE da rotina.

1. Meta o seu programa, atendendo à 3.ª nota, e incluindo a linha

   LOAD «LARGE CHARS» DATA t ()

2. MERGE « », entrada da rotina (listagem 2).
3. Carregue o programa inicialmente gravado (listagem 1), através da linha que já indicámos para incluir no seu programa.

```
9810 DATA 128,128,128,128,128,12
8,128,128,128
9811 DATA 133,138,128,133,138,12
8,132,136,128
9812 DATA 128,138,138,128,128,12
8,128,128,128
9813 DATA 141,141,136,141,141,13
6,133,133,128
9814 DATA 128,141,136,128,143,13
6,128,141,138
9815 DATA 133,138,136,128,137,12
8,129,133,138
9816 DATA 132,135,128,132,134,13
2,133,140,134
9817 DATA 128,138,128,128,128,12
8,128,128,128
9818 DATA 132,130,128,133,128,12
8,129,136,128
9819 DATA 128,129,136,128,128,13
8,128,132,130
```

```
9820 DATA 129,141,137,133,143,14
3,132,135,134
9821 DATA 128,138,128,143,143,13
8,128,138,128
9822 DATA 128,128,128,128,128,12
8,132,130,128
9823 DATA 128,128,128,133,143,13
8,128,128,128
9824 DATA 128,128,128,128,128,12
8,133,128,128
9825 DATA 128,128,137,128,137,12
8,137,128,128
9826 DATA 132,131,136,133,128,13
8,129,140,130
9827 DATA 128,141,128,128,133,12
8,128,141,136
9828 DATA 132,131,136,129,132,13
0,132,142,136
9829 DATA 132,131,136,128,133,12
8,129,140,130
9830 DATA 128,141,128,133,141,13
6,128,133,128
9831 DATA 133,131,128,129,134,12
8,132,137,128
9832 DATA 128,137,128,133,140,12
8,133,141,128
9833 DATA 129,131,138,128,137,12
8,128,138,128
9834 DATA 132,131,136,128,143,12
8,129,140,130
9835 DATA 132,131,138,129,140,13
8,128,128,138
9836 DATA 128,136,128,128,130,12
8,128,138,128
9837 DATA 128,136,128,128,130,12
8,132,130,128
9838 DATA 128,137,128,133,128,12
8,128,134,128
9839 DATA 128,128,128,129,131,13
0,129,131,130
9840 DATA 128,134,128,128,128,13
8,128,137,128
9841 DATA 132,131,136,128,137,12
8,128,136,128
9842 DATA 128,140,128,133,133,13
7,128,131,128
9843 DATA 133,131,138,133,140,13
8,133,128,138
9844 DATA 133,131,136,133,131,13
6,133,140,130
9845 DATA 132,131,136,133,128,12
8,129,140,130
9846 DATA 133,131,136,133,128,13
8,133,140,130
9847 DATA 133,131,130,133,131,13
0,133,140,136
9848 DATA 133,131,130,133,131,13
0,133,128,128
9849 DATA 132,131,136,133,132,13
6,129,140,130
9850 DATA 133,128,138,133,131,13
8,133,128,138
9851 DATA 129,139,128,128,138,12
8,132,142,128
9852 DATA 128,129,138,132,128,13
8,129,140,130
9853 DATA 133,132,130,133,138,12
8,133,129,136
9854 DATA 133,128,128,133,128,12
8,133,140,136
9855 DATA 142,132,138,138,130,13
8,138,128,138
9856 DATA 133,128,138,133,134,13
8,133,128,138
9857 DATA 132,131,136,133,128,13
8,129,140,130
9858 DATA 133,131,136,133,140,13
0,133,128,128
9859 DATA 132,131,136,133,132,13
8,129,140,134
9860 DATA 133,131,136,133,140,13
0,133,129,136
9861 DATA 132,131,136,128,134,12
8,129,140,130
9862 DATA 131,139,130,128,138,12
8,128,138,128
9863 DATA 133,128,138,133,128,13
8,129,140,130
9864 DATA 138,128,138,133,133,12
8,128,138,128
9865 DATA 138,128,138,138,136,13
8,139,129,138
9866 DATA 133,128,138,128,143,12
8,133,128,138
9867 DATA 133,128,138,129,140,13
0,128,143,128
9868 DATA 129,131,138,128,137,12
8,133,140,136
9869 DATA 132,130,128,133,128,12
8,129,136,128
9870 DATA 128,128,137,128,137,12
8,137,128,128
9871 DATA 128,129,136,128,128,13
8,128,132,130
9872 DATA 128,141,136,129,133,12
9,128,133,128
9873 DATA 128,128,128,128,128,12
8,132,140,136
9874 DATA 132,131,136,133,140,12
8,133,140,136
9875 DIM t(65,9): FOR w=1 TO 65:
 FOR u=1 TO 9: READ t(w,u): NEXT
 u: NEXT w
9880 PRINT AT 10,6;"GRAVAR ARRAY
 NUMERICO"
9885 SAVE "LARGECHARS" DATA T()
9886 PRINT AT 10,6;"VERIFICAR AR
RAY NUMERICO"
9887 VERIFY "LARGECHARS" DATA T(
)
```

LISTAGEM 2

```
9010 IF LEN z$>10 THEN RETURN
9015 DIM w$(3,30)
9020 FOR y=1 TO LEN z$
9025 IF CODE z$(y)>96 AND CODE z
$(y)<123 THEN LET z$(y)=CHR$ (CO
DE z$(y)-32)
9030 IF CODE z$(y)>96 OR CODE z$
(y)<32 THEN RETURN
9035 LET v=0
9040 FOR w=1 TO 3: FOR u=1 TO 3
9045 LET v=v+1
9050 LET w$(w)((y-1)*3+u)=CHR$ t
(CODE z$(y)-31,v)
9055 NEXT u: NEXT w: NEXT y
9060 PRINT w$(1)'w$(2)'w$(3): PR
INT
9065 RETURN
```

PROGRAMA MAT MAT

Correcções (n.º 18, pág. 11)

HUGO ASSUMPÇÃO, o autor deste programa, chama a atenção para algumas alterações:

Linha 1 será 1 PAPER 7 : BORDER 7 : INK 0 : CLS
Eliminar as linhas 620, 625 e 626

TRANSISTORES

SPECTRUM 16 e 48 K

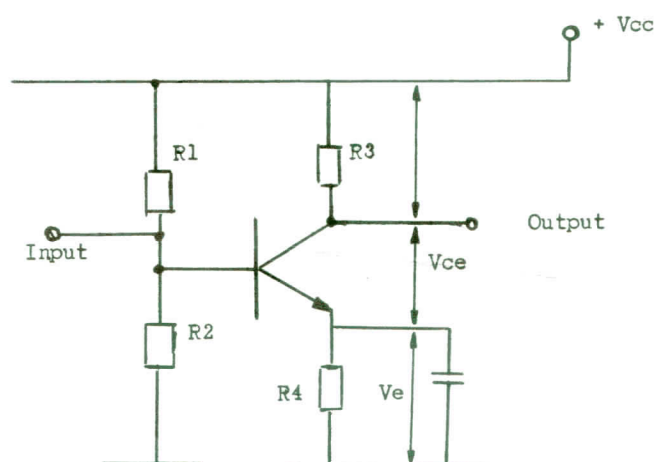
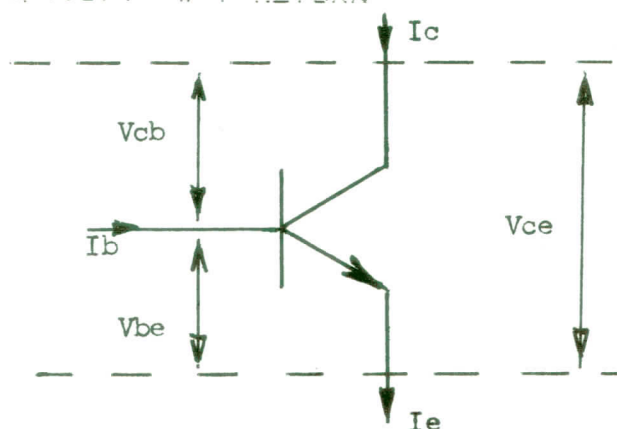
Adapt.: ALEXANDRE SOUSA

CÁLCULO DA SELECÇÃO DO PONTO ÓPTIMO DE TRABALHO

Embora saibamos que o circuito integrado substitui, na grande parte das aplicações, os grupos de transistores usados até há pouco tempo, vamos publicar um programa de cálculo para utilizar no caso do transistor individual.

Os valores Vce, Ic e Ib podem ser encontrados nas características do componente ou em tabelas de equivalencias.

O programa fornece os valores dos componentes (resistências) necessários para o bom funcionamento do circuito. Também obtém a potência de dissipação correspondente a cada resistência.

É natural que tenha de adaptar os valores obtidos aos valores standard.

```
10 CLS
11 PRINT "Seleccionar o ponto : operacao"
12 PRINT "de um TRANSISTOR a partir das "
13 PRINT "  caracteristicas do colector e"
14 PRINT "sendo possivel determinar a "
15 PRINT "Tensao BASE-EMISSOR (Vbe) a par- tir dos dados do transistor"
20 PRINT : PRINT "Se Vbe nao e conhecido pode par-tir do valor ... 0.6 V p/ trans.Silicio e 0.3 V p/trans.Germanio"
22 PRINT : PRINT "O valor tipico p/tensao Ve     ": PRINT "(Emissor/Gnd... 1 V"
24 PRINT : PRINT "Dar entrada dos valores..."
30 INPUT "Colector/Emissor...(V) ";vce,,,"Corr. colector....(mA) ";ic,,,"Corr.Base.........(uA) ";ib,,,"Base/Emissor.......(V) ";vbe,,,"Emissor/(tensao)...(V) ";ve
40 CLS : PRINT TAB 3;" Operacao do transistor": PRINT TAB 4;"______________________________"
50 PRINT "Colector/Emissor =";TAB 20;vce;TAB 31;"v"
60 PRINT "Corrente -Colector :"; TAB 20;ic; TAB 30;"ma"
70 PRINT "Corrente - Base =";TAB 20;ib;TAB 30;"ua"
80 PRINT "Base-Emissor =";TAB 20;vbe;TAB 31;"v"
90 PRINT "Emissor/GND =:";TAB 20;ve;TAB 31;"V"
100 LET ib=ib/1000: LET ip=10*ib: LET vp=vbe+ve: LET vbb=2*vce+ve: LET it=ic+ip
110 LET r1=(vbb-vp)/ip: LET r1=INT (r1/.001+.5)
120 LET r2=vp/(ip-ib): LET r2=INT (r2/.001+.5)
130 LET r3=(vbb-vce-ve)/ic: LET r3=INT (r3/.001+.05)
140 LET r4=ve/(ic+ib): LET r4=INT (r4/.001+.5)
150 LET p=(vbb-vp)*ip/1000: GO SUB 300: LET a$=p$
160 LET p=vp*(ip-ib)/1000: GO SUB 300: LET b$=p$
170 LET p=(vbb-vce-ve)/1000: GO SUB 300: LET c$=p$
180 LET p=ve*(ic+ib)/1000: GO SUB 300: LET d$=p$
190 PRINT : PRINT "Resistencia (Ohm)...."
200 PRINT TAB 3;"R1";TAB 16-LEN STR$ r1;r1;TAB 24;a$
210 PRINT TAB 3;"R2";TAB 16-LEN STR$ r2;r2;TAB 24;b$
220 PRINT TAB 3;"R3";TAB 16-LEN STR$ r3;r3;TAB 24;c$
230 PRINT TAB 3;"R4";TAB 16-LEN STR$ r4;r4;TAB 24;d$
240 PRINT : PRINT "F.Alim.=";vbb;" V / ";it;" mA"
250 PRINT AT 21,0;;"Enter para novo calc. ou STOP"
260 PAUSE 0: IF CODE INKEY$=13 THEN CLS : GO TO 24
270 IF CODE INKEY$=226 THEN STOP
280 GO TO 260
300 REM potencia
310 IF p<=0.125 THEN LET p$="0.125  W": RETURN
320 IF p<=0.25 THEN LET p$="0.25  W": RETURN
330 IF p<=0.5 THEN LET p$="0.5  W": RETURN
340 IF p<=1 THEN LET p$="1   W": RETURN
350 IF p<=2.5 THEN LET p$="2.5  W": RETURN
360 IF p<=5 THEN LET p$="5   W": RETURN
370 IF p>5 THEN LET p$=STR$ INT (p+.5)+" W": RETURN
```

NOVOS PROGRAMAS — SPECTRUM

JOGOS

CÓDIGO*/PREÇOS

- APOCALYPSE (48 K) — Um jogo de estratégia que exige muita perícia e lógica. Um jogador terá de escolher uma época histórica e possui 3 meios de ataque/defesa: exército, armada e mísseis nucleares. O jogo pode decorrer num período pré-nuclear se todos os jogadores forem de acordo e tomarão então o papel de líderes do mundo. **IP/600$00**
- DOOMSDAY CASTLE (48 K) — Neste castelo habita SCARTHAX, um ser demoníaco que esconde as suas pedras mágicas. A ajudá-lo há umas criaturas estranhas (URKS) que libertam radiações "theta" e outros seres também perigosos (GARTHROGS, ORPHCS, NEUCLOIDS, GOOGLY BIRDS). Você possui uma cápsula auto-controlada, um míssil e um campo magnético e somará pontos conforme as pedras que conseguir. **IP/400$00**
- EVOLUTION (48 K) — Imagine-se na terra há 3 500 anos atrás: vulcões em erupção, céu carmesim, sol quentíssimo, falta de oxigénio . . . Ocorrem assim os primeiros acontecimentos da origem do mundo e da vida. Começa a evolução, cujas fases você vai seguir até ao aparecimento do homem. **IP/400$00**
- FIGHTER PILOT (48 K) — Simulação de um voo baseado num F15 da Força Aérea dos E.U.A., com 8 opções: aterragem, voo de treino, prática de 1 combate aéreo, combate aéreo, aterragem c/ nevoeiro, voos c/ ventos contrários, pilotagem de perícia, controlo. Gráficos 3 D e compatibilidade c/ Joystick (Kempston). **I/400$00**
- PINBALL (16 K) — Simulação das máquinas "flippers". Lance a bola e esforce-se por não a perder, de modo a atingir o máximo de pontuação. **IP/400$00**
- SPACE SHUTTLE (48 K) — simulação de um voo espacial, c/ gráficos de alta resolução — indicadores c/ todas as informações necessárias. Pilote a sua nave e coloque-a na posição ideal para recolher 1 satélite à deriva. Depois aterre sem motores no deserto das Areias Brancas e, no final, ficará a saber se os seus passos foram ou não correctos e a pontuação atribuída. **I/400$00**

EDUCAÇÃO

- ALGB SEC (48 K) — Equações do 2.° grau (9.° ano unificado). Trinómio do 2.° grau (11.° ano). Números complexos: operações na forma algébrica e na forma trigonométrica. **P/600$00**
- BIOLOGIA (48 K) — Contém sub-programas com: revisões de matérias pertinentes c/ cerca de 100 diagramas e textos complementares. Problemas para determinação de genotipos. Questões de escolha múltipla (peça a solução se não acertar). Natureza de substâncias. Esclarecimentos sobre diagramas. Alusões a experiências científicas. Algumas das matérias: Aparelhos circulatório, digestivo, respiratório . . ., reprodução, metabolismo, sistema nervoso, hereditariedade, nutrição, evolução e selecção natural das espécies, classes, fotosíntese, etc. . . . **I/600$00**
- FRACÇÕES (48 K) — Máximo divisor comum; mínimo múltiplo comum; operações c/ fracções — 2.° ano (Ciclo). **P/600$00**
- FUNÇÕES (48 K) — Gráficos e funções nas formas: y = f(x); x = x(t) e outras — Ensino Superior. **P/600$00**
- GEOMETRIA DESCRITIVA (48 K) — Toda a nomenclatura c/ apresentação gráfica de planos, intercepções, etc. Possui 1 teste sobre a situação dos pontos em relação aos planos de projecção — 10.° ano. **P/600$00**
- HOMOTETIA (48 K) — Apoio ao ensino da homotetia (teoria e prática) — 8.° ano. **P/600$00**
- LÓGICA (48 K) — Construção de tabelas de verdade c/ duas e três variáveis — 10.° ano. **P/600$00**
- FÍSICA (48 K) — Contém sub-programas c/: revisões de matérias essenciais, c/ cerca de 250 diagramas e textos. Problemas c/ gráficos e cálculos matemáticos. Equações de escolha múltipla. Alguns temas: Leis da física, mecânica, electricidade, magnetismo, pressão, luz, electrostática, radioactividade, etc. **I/600$00**
- RECTA (48 K) — As diversas formas de definir uma recta e as correspondentes equações (teoria e prática) — 10.° ano. **P/600$00**
- SUC, f(x) (48 K) — Limites de vários tipos de sucessões. Estudo de funções f(x) e seus gráficos — 11.° e 12.° anos. **P/600$00**
- TR GEOM (48 K) — Apoio ao ensino de translações, rotações, simetrias axiais (teoria e prática) — 7.° ano. **P/600$00**

UTILITÁRIOS

- THE KEY (16/48 K) — Faz a cópia de outros programas **I/1 000$00**

* **Códigos:** P — programa e instruções em português.
I — programa e instruções em inglês.
IP — programa em inglês e instruções em português.

DESCONTO DE 20 % PARA SÓCIOS — VENDAS NA SEDE DO CLUBE OU PEDIDOS À COBRANÇA

??? MERCADO Z80 ???

Lembram-se da proposta do sócio José Gorda?
Aproveitando as suas ideias, o MERCADO Z80 será uma secção do CLUBE Z80 que **empresta aos seus sócios PROGRAMAS e LIVROS** para consulta e melhor conhecimento / aproveitamento de microcomputadores.
Depois de analisarmos pormenorizadamente as implicações desta iniciativa, excluindo a hipótese de os Sócios do Mercado (S. M.) copiarem e fotocopiarem programas e livros, optámos por estruturar o MERCADO Z80 do modo que nos parece mais funcional.
Assim, as «normas» que apresentamos em baixo parecem-nos, à partida, concretizáveis. Contudo, dado que não prevemos qual será a adesão a esta iniciativa (apenas 6 sócios se pronunciaram favoravelmente), não sabemos também se teremos as condições necessárias para responder aos interessados. Referimo-nos essencialmente ao tempo que será necessário para deslocações aos CTT's, controlo de materiais em circulação, controlo de «contas correntes», etc..
Efectivamente, se o número de aderentes for grande, seremos obrigados a empregar uma pessoa exclusivamente nesta tarefa, o que implicará outros problemas...
Por isso, o CLUBE Z80 faz-lhe um apelo: Depois de tomar conhecimento das «normas» do MERCADO Z80, se estiver interessado em «entrar» nele, confirme-nos por escrito até ao dia 16 de Maio. P.F. anexe um envelope selado e endereçado a si próprio — o CLUBE Z80 informá-lo-á da data de abertura do MERCADO e do n.º de aderentes. Para seu próprio interesse, comunique-nos também a sua opinião quanto ao regulamento do MERCADO Z80 que apresentamos a seguir:

1 — O MERCADO Z80 funciona exclusivamente para sócios do CLUBE Z80.

2 — Os produtos que o MERCADO Z80 empresta:
- — Livros anunciados no Clube Z80 (há 1 exemplar de cada).
- — Programas de **JOGOS** existentes no CLUBE Z80 (há 5 exemplares de cada).

3 — O MERCADO Z80 funciona apenas por correspondência (CTT), mesmo para sócios residentes no Porto. (As instalações do CLUBE e o pessoal disponível não permitem o atendimento directo).

4 — Cada sócio terá a sua «CONTA-CORRENTE» que começará com um depósito de Esc.: 1000$00 no MERCADO Z80.

5 — Por cada encomenda enviada, o MERCADO Z80 cobrará ao sócio uma taxa de Esc.: 100$00.

6 — Na «conta-corrente» do sócio, por cada pedido, debitar-se-ão 3 tipos de despesas:
- — Embalagem Postal (20$00 a 30$00).
- — Portes dos CTT's.
- — 100$00 (taxa de utilização dos produtos).

NOTA: Estas despesas não serão cobradas directamente pelos CTT's, pois esse processo agrava os custos. Contudo, cada sócio pode optar e informar-nos como prefere.

7 — Quando a «Conta-Corrente» deixar de ter saldo positivo, o S. M. deverá renovar o s/ depósito (1000$00). Ele próprio deverá controlar a sua «C.C.» pois nem sempre o CLUBE Z80 tem hipóteses de o avisar.

8 — Para além da «conta-corrente, o S.M. depositará no MERCADO Z80 a quantia de Esc. 2 000$00 para salvaguardar o caso de não devolver o material pedido. Essa quantia será devolvida ao S.M. quando ele desistir do MERCADO Z80.

9 — **Limites de produtos pedidos**

O seu pedido nunca pode ultrapassar 5 UNIDADES (entre livros e cassettes) de cada vez. Quanto a livros, não serão emprestados mais que dois de cada vez (só há 1 exemplar de cada!).

Ex.: 2 livros + 3 cassettes
1 livro + 4 cassettes
5 cassettes

10 — **Tempo para utilização**

Os produtos recebidos poderão ficar nas mãos do sócio durante 15 dias, findos os quais deverão ser devolvidos ao MERCADO Z80 em ESTADO DE CONSERVAÇÃO e FUNCIONAMENTO idêntico àquele em que foram enviados.

NÃO ESQUEÇA: Confirme até 16 de Maio a sua adesão ao MERCADO Z80.

PROGRAMA MERGE ECRAN

(n.º 17, pág. 7)

- HUGO ASSUMPÇÃO pergunta: "Não há possibilidade de transformar as linhas 100 a 180 em código-máquina? É que o transporte do ecran para a memória, em BASIC, de mora cerca de 3 minutos. Ao passo que o inverso é imediato (mas a preto e branco, visto que os atributos não são transferidos). Se sim, como? Usando a subrotina apresentada no n.º 10, pág. 6?"
- FERNANDO PRECES, o autor de MERGE ECRAN, responde: "Este programa foi escrito para simples demonstração, procurando cativar os leitores para a linguagem-máquina. A rotina em BASIC pode, de facto, ser substituída e aqui vai a respectiva alteração. Acrescente:

```
 62 LET x = 32722
 65 FOR n = x TO x + 12
 70 READ b: POKE n,b
 75 PRINT n,PEEK n
 80 NEXT n
 90 REM *****
100 REM Transporte do ecran para a memória, progra-
    mado em cód.-máq.
105 REM (GO TO 110)
110 RANDOMIZE USR 32722
120 STOP
215 REM (GO TO 220)
310 DATA 33,0,64,17,144,101,1,0,24,237,176,201,0
```

Como pode verificar, a rotina em C.M. é muito semelhante à da pág. 6 do boletim 10, c/ pequenas diferenças pelo facto de ter sido escrita para o ZX81."

CLUBE Z80

INSCRIÇÃO COMO ASSOCIADO

O **CLUBE Z80** está aberto a todos os utilizadores de microcomputadores.

A intenção de associar os entusiastas das micro-máquinas, é exclusivamente a de permitir:

1 — PUBLICAÇÃO DE UM JORNAL MENSAL, onde sejam publicados programas de uso geral ou específico como no caso da educação.

2 — PROMOVER TROCAS DE PROGRAMAS, e trocas de experiências; tanto no caso do Software (programação), como no caso do Hardware (electrónica).

3 — PROMOVER DESCONTOS NA AQUISIÇÃO DE PROGRAMAS.

4 — LANÇAR CURSOS DE PROGRAMAÇÃO EM BASIC — PASCAL OU OUTRAS LINGUAGENS E DIVULGAR O USO DE LINGUAGEM MÁQUINA.

NOME ....

IDADE .... COMPUTADOR TIPO ....

PROFISSÃO ....

ENDEREÇO ....

.... TELEF. ....

ASSINATURA ANUAL — Esc. 1 500$00 ☐

ASSINATURA SEMESTRAL — Esc. 750$00 ☐

CHEQUE OU VALE DO CORREIO

N.º ....

BANCO ....

DATA ....../....../......

JÁ SÓCIO ☐

NOVO SÓCIO ☐ → A partir do mês de .... (inclusive)